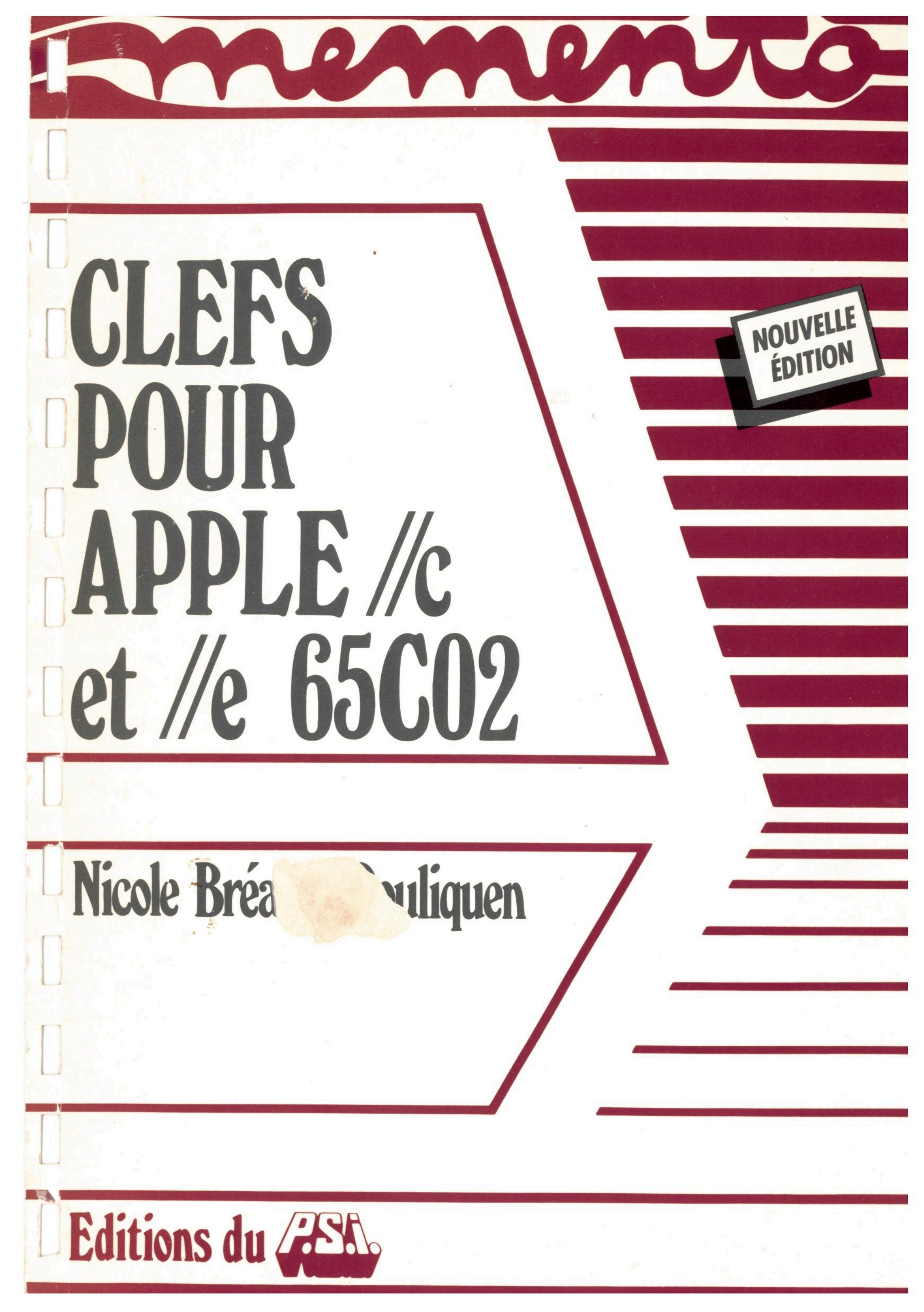

mémento
CLEFS POUR APPLE //c et //e 65C02
NOUVELLE ÉDITION
Nicole Bréa uliquen
Editions du P.S.I.

CLEFS POUR APPLE //c et //e 65C02

Connaissez-vous la collection système Apple chez P.S.I. ?

Autres ouvrages relatifs à l'Apple II

Maîtrise du Basic

— L'Apple et ses fichiers — Jacques Boisgontier
— Basic Plus, 80 routines sur Apple II — Michel Martin

Assembleur

— Introduction à Prodos sur Apple — Francis Versheure
— Clefs pour Apple II — Nicole Bréaud-Pouliquen
— Programmation système de l'Apple II — Marcel Cottini
— Assembleur de l'Apple II — Nicole Bréaud-Pouliquen et Daniel-Jean David

Ouvrages contenant des programmes en assembleur sous D.O.S. 3.3 ou ProDOS

— La programmation des jeux d'arcades sur Apple II — Jean-Luc Fisher
— Apple, modems et serveurs — Alain Mariatte
— Création et animations graphiques sur Apple II — Gilles Fouchard et Jean-Yves Corre (livre + disquette)

A paraître :

— Système ProDOS sur Apple II — Marcel Cottini

Pour tout problème rencontré dans les ouvrages P.S.I.
vous pouvez nous contacter au numéro ci-dessous :

Numéro Vert/Appel Gratuit en France

05 21 22 01

(Composer tous les chiffres, même en région parisienne)

ISBN : 2-86595-329-7

CLEFS POUR APPLE //c et //e 65C02

par

Nicole Bréaud-Pouliquen



Éditions du PSI
1986

Nicole BREAUD-POULIQUEN *est ingénieur-conseil en informatique individuelle. Dans ce cadre, elle enseigne la programmation depuis plusieurs années.*

Elle pratique l'APPLE II depuis sa commercialisation en France et s'intéresse plus particulièrement à l'insertion des ordinateurs individuels dans l'enseignement.

Du même auteur :

Aux Editions du P.S.I.

- La Pratique de l'Apple II - volume I
- La Pratique de l'Apple II - volume II
- L'Assembleur de l'Apple II
- Du LOGO pour Apple II
- Les Ressources de l'Apple //c

PRESENTATION

Ce livre est destiné à se trouver en permanence à côté de votre Apple IIc lorsque vous l'utilisez. Il renferme toutes les informations de référence que vous pouvez souhaiter retrouver rapidement : syntaxe des commandes, codes caractères, messages d'erreur, langage machine et adresses du système.

Les informations sont données sans détails excessifs, le but principal de ce livre étant l'accès rapide à l'information : pour un exposé plus introductif et plus complet, vous pouvez vous reporter au livre "Les Ressources de l'Apple IIc".

Le chapitre "comment...?" rassemble astucieusement tous les "trucs" de différents niveaux qu'il est utile de savoir.

Nous serons reconnaissants à nos lecteurs de nous suggérer toute information complémentaire à incorporer aux futures éditions de ce livre.

SOMMAIRE

FONCTIONS DU BASIC APPLESOFT

Une fonction demande un argument (ou plusieurs) et renvoie une valeur qui est le résultat de l'application de cette fonction à la valeur de l'argument.

Fonctions mathématiques

ABS Valeur absolue de l'argument entre parenthèses.

ATN Arc tangente - le résultat est en radians, compris entre $-\pi/2$ et $+\pi/2$.

COS Cosinus - l'argument doit être en radians. Exemple : cos(x en degrés) ⇒ COS(π*X/180).

EXP Exponentielle e^X. L'argument doit être ≤ 88 sinon il se produit un dépassement de capacité.

INT Partie entière, plus exactement le plus grand entier inférieur ou égal à l'argument : INT(0.5) vaut 0 ; INT(5) vaut 5 ; INT(-0.5) vaut -1 ; INT(-3) vaut -3.

LOG Logarithme naturel (népérien ou en base e). Pour obtenir le logarithme de X en base Y, utiliser LOG(X)/LOG(Y). Exemple : logarithme décimal de X⇒LOG(X)/LOG(10).

RND Fournit un nombre pseudo-aléatoire supérieur ou égal à 0 et inférieur à 1, avec un argument positif.
Exemple : PRINT RND(1) ⇒.103112573
Si les appels de la fonction se succèdent, les résultats suivront toujours la même suite de nombres aléatoires. Un appel de la fonction avec un argument négatif permet d'amorcer une suite particulière. RND(0) renvoie le dernier nombre généré.

SGN Fonction "signe" : 1 si X>0, -1 si X<0 et 0 si X=0.

SIN Sinus - l'argument est supposé en radians.

SQR Racine carrée - l'argument doit être supérieur ou égal à 0.

TAN Tangente - l'argument est supposé en radians.

Fonctions de tabulation

POS POS(Ø) fournit la prochaîne position d'affichage libre sur la ligne d'écran (position horizontale du curseur). Les valeurs retournées vont de Ø à 79 (80 colonnes) et de Ø à 39 (40 colonnes).

SPC Ne peut s'employer que dans une instruction PRINT. PRINT SPC(X); imprime X espaces. X doit être entier compris entre Ø et 255.

TAB Ne peut s'employer que dans une instruction PRINT TAB(X). Fait aller à la position d'impression n°X (1 est la position la plus à gauche d' une ligne, 40 la plus à droite), X doit être compris entre 1 et 255 TAB(Ø). Déplace le curseur à la position 256. Si X<position courante du curseur, alors il n'y a pas d'action. TAB ne déplace jamais le curseur vers la gauche.

HTAB Positionne le curseur horizontalement avant l'instruction PRINT. HTAB 1 correspond à la position la plus à gauche de la fenêtre d'écran. La position extrême est 255 (5 lignes plus loin).

VTAB Positionne le curseur verticalement avant l'instruction PRINT. VTAB 1 correspond à la 1ère ligne du haut de l'écran quelle que soit la valeur de la marge haute de la fenêtre.

VTAB 24 positionne à la dernière ligne d'écran. Si l'argument est supérieur à la marge basse de la fenêtre d'écran, alors l'affichage ne pourra se faire qu'à la ligne pointée par l'argument pour toutes les instructions PRINT suivantes.

Fonctions système

FRE Quelle que soit la valeur de l'argument, fournit le nombre d'octets restés libres en mémoire. Provoque un nettoyage des chaînes abandonnées.

PEEK Fournit le contenu (compris entre Ø et 255) de la case mémoire dont l'adresse est égale à son argument (qui doit être entier et compris entre Ø et 65 535).

USR Appel d'un programme utilisateur en langage machine. L'unique argument est transmis à travers l'accumulateur flottant. L'adresse du sous-programme doit être préenregistrée en $ØB et $ØC avec JMP ($4C) en $ØA. Le résultat est placé dans l'accumulateur flottant, comme pour toutes les fonctions de calcul.

Fonctions chaînes de caractères

LEN(X$) Longueur (de Ø à 255).

LEFT$(X$,N) Extraction des N caractères les plus à gauche.

RIGHT$(XS,N) Extraction des N caractères les plus à droite.

MID$(X$,K) ou **MID$(X$,K,N)** Extraction au milieu de tous ou de N caractères à partir de la kème position. K doit être égal ou supérieur à 1.

Fonction de conversion

ASC(X$) Renvoie le code ASCII du premier caractère de la chaîne X$. ASC("A") vaut 65.

CHR$(K) Renvoie le caractère dont le code ASCII vaut K. CHR$(4) est 'Ctrl D'.

STR$(A) Donne la représentation d'un nombre en chaîne de caractères à partir de sa valeur numérique A.

VAL(X$) Donne la valeur numérique représentée par la chaîne X$.

Fonctions graphiques (basse résolution)

COLOR= Donne une couleur (Ø à 15) pour le prochain tracé en basse résolution.

PLOT X,Y Place un petit domino à l'abscisse X et à l'ordonnée Y. X et Y vont de Ø à 39. Ø,Ø est la position en haut, à gauche.

HLIN X1,X2 AT Y Trace une ligne horizontale entre X1 et X2 à l'ordonnée Y.

VLIN Y1,Y2 AT X Trace une ligne verticale entre Y1 et Y2 à l'abscisse X.

SCRN(X,Y) Renvoie la couleur du domino tracé en X,Y. Renvoie le code-écran du caractère de texte avec la formule SCRN(x,2*y)+16*SCRN(x,2*y+1), x entre Ø et 39 et y entre Ø et 23.

Fonctions graphiques (haute résolution)

HCOLOR= Fixe la couleur (Ø,1,2,3) (noir, vert, violet, blanc) (4,5,6,7) (noir, orange, bleu, blanc) du prochain point à tracer. Les couleurs Ø,2,4,6 ne peuvent être attribuées qu'aux points situés dans des colonnes paires. Les couleurs Ø,1,5,7 ne peuvent être attribuées qu'aux points situés dans des colonnes impaires.

HPLOT X,Y — Place un point à l'abscisse X et l'ordonnée Y. X va de Ø à 279. Y de Ø à 159 (HGR) ou à 191 (HGR2). Ø,Ø est en haut et à gauche.

HPLOT X1,Y1 TO X2,Y2 — Trace une ligne entre deux points, la commande peut être étendue à d'autres points... TO XN,YN.

DRAW F AT X,Y — Dessine la forme n° F de la table des formes en commençant au point X,Y, de la couleur HCOLOR ; la forme efface le fond.

XDRAW F AT X,Y — Dessine la forme n° F de la table des formes. La couleur de chaque point est le complément de la couleur actuelle du point sur l'écran. Si la couleur du fond est noire, XDRAW dessine en blanc. En redessinant par XDRAW, le fond est restauré.

ROT= — L'argument est proportionnel à un angle de rotation à faire subir à la forme à dessiner par DRAW. ROT=16 correspond à 90°.

SCALE= — Donne une valeur d'agrandissement à la forme à dessiner de 1 à 255. A chaque incrément de SCALE, on double le nombre d'angles possibles de ROT.

Fonctions : manettes de jeux

PDL(X) — Renvoie un nombre de Ø à 255 proportionnel à la position angulaire de la manette (potentiomètre). X vaut Ø ou 1.
Si la souris est ne ligne (grâce à la séquence d'instructions indiquées plus loin), PDL(Ø) renvoie la position horizontale et PDL(1) renvoie la position verticale.

PEEK(X-16287) — Donne un résultat>127 si le bouton poussoir n° X a été pressé. X vaut Ø ou 1.
Le bouton Ø est aussi la touche "".
Le bouton 1 est aussi la touche "".

Fonction haut-parleur

PEEK(-16336) — Emet un "clicks' par le haut-parleur (le volume des sons est réglable manuellement sur le côté gauche du clavier).

Les caractères de contrôle et d'échappement

Ils servent à réaliser des opérations spécifiques soit pendant l'exécution, soit pour l'affichage, soit pour l'impression, soit pour activer ou désactiver des modes particuliers.

En tapant au clavier 'Control/x', la lettre ou le symbole x ne s'affiche pas mais l'effet aura bien lieu.

Il faut distinguer les caractères dont l'effet ne se produira que s'ils sont envoyés par une instruction PRINT (mode programmé) et ceux qui produisent des effets dès qu'ils sont tapés au clavier (mode direct).

L'Apple //C est conçu pour l'affichage des textes sur 80 colonnes de large et contient donc des sous-programmes (en MEM) de gestion de cet affichage ; ces sous-programmes doivent être activés explicitement.

En effet, au démarrage, le système Apple //C se prend pour un Apple II ou //e en ce qui concerne le mode d'affichage ; il n'affiche que sur 40 colonnes et ne sait pas encore réagir à des caractères de contrôle sophistiqués. Pour donner à l'Apple //C tous ses nouveaux moyens, il est nécessaire d'activer les sous-programmes d'affichage en 80 colonnes. Nous indiquons par *, dans le tableau ci-dessous, quels sont les caractères de contrôle gérés par les sous-programmes d'affichage en 80 colonnes.

Actions demandées	*Par le clavier*	*Par PRINT...*
Arrêt de l'exécution d'un programme à tout moment.	Control C	non
Annulation de la saisie en cours au clavier.	Control X	non
Emission d'un bip sonore.	Control G	CHR$(7)
Envoi d'un retour au début de la ligne suivante.	Control M ou Return	CHR$(13)
Suspension de l'affichage et reprise du défilement par n'importe quelle touche.	Control S	non
Affichage en 80 colonnes (activation des sous-programmes 80 colonnes).	Escape 8	* CHR$(18)
Affichage en 40 colonnes (sous-programmes 80 colonnes activés).	Escape 4	* CHR$(17)
Désactivation des sous-programmes d'affichage en 80 colonnes.	* Escape Control Q	* CHR$(21)
Inhibition des caractères Ctrl.	Escape Control D	* CHR$(4)
Autorisation des caractères Ctrl à l'exception de G,H,J,M.	Escape Control E	* CHR$(5)

Actions demandées	*Par le clavier*	*Par PRINT...*
Effacement de l'écran et curseur en haut à gauche.	Escape à	* CHR$(12)
Déplacement du curseur.	Escape	
Curseur une ligne plus bas.	Escape C ou Ctrl J	CHR$(1Ø)
Curseur une ligne plus haut.	Escape D	* CHR$(31)
Curseur un cran à droite.	Escape A	* CHR$(28)
Curseur un cran à gauche.	Escape B ou Ctrl H	CHR$(8)
Déplacement prolongé vers le haut.	(Escape) I ou ↑ répétés	non
Déplacement vers le bas.	(Escape) M ou ↓ répétés	non
Déplacement vers la gauche.	(Escape) J ou ← répétés	non
Déplacement vers la droite.	(Escape) K ou → répétés	non
Renvoi du curseur en haut à gauche sans effacement.	non	* CHR$(25)
Effacement jusqu'à la fin de la ligne.	Escape E	* CHR$(29)
Effacement jusqu'au bas de l'écran.	Escape F	* CHR$(11)
Effacement de la ligne où se trouve le curseur.	non	* CHR$(26)
Décalage de toute l'image d'une ligne vers le bas, le curseur ne bouge pas.	non	* CHR$(22)
Décalage de toute l'image d'une ligne vers le haut, le curseur ne bouge pas.	non	* CHR$(23)
Affichage en normal.	non	* CHR$(14)
Affichage en inversé.	non	* CHR$(15)
Affichage des icones au lieu des majuscules inversées.	non	* CHR$(27)
Affichage des majuscules inversées au lieu des icones.	non	* CHR$(24)

Liste des caractères de contrôle par ordre croissant des codes ASCII

Code en décimal	*Nom*	*Action*
0	**Ctrl à**	Aucune en Basic.
1	**Ctrl A**	Réservé au Modem ou à la Souris.
2	**Ctrl B**	Aucune en Basic.
3	**Ctrl C**	Arrêt du programme en Basic.
4	**Ctrl D ***	Précédé de Esc, inhibe les caractères Ctrl.
5	**Ctrl E ***	Précédé de Esc, autorise les caractères Ctrl.
6	**Ctrl F**	Aucune en Basic.
7	**Ctrl G**	Bip ou "bell".
8	**Ctrl H**	Recul du curseur ou "backspace".
9	**Ctrl I**	Réservé à l'imprimante.
10	**Ctrl J**	Saut de ligne ou "LF".
11	**Ctrl K ***	Effacement jusqu'en bas ou "CLREOP".
12	**Ctrl L ***	Effacement de tout l'écran ou "HOME".
13	**Ctrl M**	Retour au début de la ligne suivante ou "CR".
14	**Ctrl N ***	Mode normal, blanc sur noir.
15	**Ctrl O ***	Mode inversé, noir sur blanc.
16	**Ctrl P**	Aucune en Basic.
17	**Ctrl Q ***	Affichage sur 40 colonnes ou "SET40".
18	**Ctrl R ***	Affichage sur 80 colonnes ou "SET80".
19	**Ctrl S**	Affichage suspendu ou "stop-list".
20	**Ctrl T**	Aucune en Basic.
21	**Ctrl U ***	Désactivation des sous-programmes pour 80 colonnes ou "QUIT".
22	**Ctrl V ***	Ecran décalé vers le bas ou "SCROLLDOWN".
23	**Ctrl W ***	Ecran décalé vers le haut ou "SCROLLUP".
24	**Ctrl X ***	Désactivation des icones ou "MOUSOFF".
25	**Ctrl Y ***	Curseur en haut à gauche ou "HOMECUR".
26	**Ctrl Z ***	Effacement de la ligne ou "CLRLIN".

Code en décimal	*Nom*	*Action*
27	**Ctrl ° ***	Activation des icones ou "MOUSON".
28	**Ctrl ç ***	Curseur avance ou "NEWADV".
29	**Ctrl § ***	Effacement jusqu'en fin de ligne ou "CLREOL".
3Ø	**Ctrl ^**	Aucune en Basic.
31	**Ctrl _ ***	Curseur va à la ligne au dessus ou "UP".

Mode obligatoire	*Mot-clé*	Définition - *exemples*
	&	Fait démarrer l'exécution d'un sous-programme en langage machine dont l'adresse figure en $3F6, $3F7 avec JMP ($4C) dans $3F5 *&16→1Ø* si le sous-programme convertit du décimal en hexadécimal.
	CALL	Fait démarrer l'exécution d'un sous-programme en langage machine à l'adresse indiquée. *CALL -151* Un argument négatif correspond au complément à 65536 de l'adresse cherchée.
	CLEAR	Remise à zéro de toutes les variables. Les chaînes ont leur longueur nulle.
Direct	**CONT**	Continuer dans le programme après interruption de l'exécution par 'Ctrl C'.
	DATA	Définit une liste de constantes qui seront "lues" par une instruction READ *1Ø DATA ABC,5,Ø.15*
	DEL	Avec deux arguments séparés par une virgule, délimite une partie de programme à supprimer *DEL 1Ø,5Ø* supprime les instructions de 1Ø à 5Ø.
	DEF FN	Définition d'une fonction utilisateur à un seul argument : *1Ø DEF FN F(X)=X-256*INT(X/256)*
	DIM	Dimensionnement d'un tableau (fixe le nombre et les valeurs maxima des indices) *1Ø DIM A(1ØØ),B%(5ØØ),C$(1Ø)* *2Ø DIM T(N)* *3Ø DIM M(1Ø,1Ø,1Ø)* 88 indices au plus.
Programmé	**END**	Arrêt d'exécution de la suite d'instructions.

Mode obligatoire	*Mot-clé*	Définition - *exemples*
	FOR	Introduit une boucle : toutes les instructions comprises entre *FOR I=A TO B STEP C* et le NEXT I seront répétées pour toutes les valeurs de I allant de A à B, C par C *10 FOR I=1 TO 1000* *20 FOR X=1.5 TO 2 STEP.1* *30 FOR J=N TO -N STEP -2* Si plusieurs boucles se succèdent avec le même indice, ne pas interrompre la progression de l'indice jusqu'à sa valeur maxima *10 FOR I=1 TO 100* *20 IF N$(I)=X$ THEN T=I:I=100* *30 NEXT I* *40 IF T=0 THEN PRINT"NON TROUVE":END* *50 PRINT"TROUVE EN";T*
	FLASH	Affichage des caractères en mode clignotant. Ce mode n'est pas neutralisé par 'Reset'. La touche → de recopie va modifier les caractères sur l'écran. Faire NORMAL pour rétablir la situation.
Programmé	**GET**	Saisie d'1 caractère au clavier, il n'est pas affiché. 'Ctrl C' n'a aucun effet *10 GET A$* n'est pas recommandé avec des instructions du SED dans le programme. Sauf si 'Ctrl D' est précédé de 'Return' *D$=CHR$(13)+CHR$(4)*
	GOSUB	Appel d'un sous-programme *10 GOSUB 1000*
	GOTO	Saut à une autre instruction numérotée *10 GO TO 50*
	GR	Met une partie de l'écran en affichage graphique de 40 x 40 dominos et l'efface. Laisse 4 lignes de texte dans le bas.

Mode obligatoire	*Mot-clé*	Définition - *exemples*
	HGR	Met une partie de l'écran en affichage graphique de 280 x 160 et l'efface Laisse 4 lignes de texte.
	HGR2	Met tout l'écran en graphique de 280 x 192 et l'efface. (Le curseur disparaît).
	HIMEM:	Spécifie la plus haute position de mémoire vive, disponible au programme.
	HOME	Efface la fenêtre d'écran et positionne le curseur en haut et à gauche de cette fenêtre. Précédée de TEXT : tout l'écran est effacé.
	IF	Saut conditionnel, de la forme IF condition THEN instruction. Si la condition n'est pas satisfaite (résultat faux ou Ø) on passe à la ligne suivante ; si la condition est satisfaite, on effectue l'instruction qui suit THEN IF C THEN GOTO x s'écrit aussi IF C THEN x ou encore IF C GOTO x *1Ø IF A>B THEN Y=K* *2Ø IF A$=""THEN 5* *3Ø IF A<Ø OR A>1ØØ THEN 1ØØ*
Programmé	**INPUT**	Acquisition de données au clavier *1Ø INPUT A* *2Ø INPUT A,B,C$,D* *3Ø INPUT"VOTRE NOM?";N$*
	IN # **ou** **IN £**	Connecte le périphérique branché sur le connecteur indiqué comme argument, en entrée du micro-ordinateur.
	INVERSE	Provoque l'affichage des caractères en noir sur blanc. Pour revenir en blanc sur noir faire NORMAL.
	LET	Est l'instruction d'affectation de valeur à une variable. N'est pas obligatoire *LET X$="AOUT"*
	LIST	Liste du programme *LIST tout le programme* *LIST 1Ø,1ØØ de 1Ø à 1ØØ* *LIST 1ØØ, de 1ØØ à la fin* *LIST, 1Ø jusqu'à 1Ø* La virgule peut être remplacée par -

Mode obligatoire	*Mot-clé*	Définition - *exemples*
	LOMEM:	Spécifie la plus basse position de mémoire vive disponible pour les variables du programme.
Direct	**NEW**	Simule l'effacement du programme actuellement en mémoire vive. (2 pointeurs sont mis à zéro).
	NEXT	Fait passer à l'itération suivante dans un FOR *NEXT I NEXT J,I NEXT*
	NORMAL	Rétablit l'affichage sur écran en blanc sur noir.
	NOTRACE	Déconnecte le mode TRACE .
	ON	ON I GOTO 1Ø,2Ø,3Ø Si I vaut 1, on va en 1Ø, s'il vaut 2, on va en 2Ø, en 3Ø s'il vaut 3 Si I est nul ou faux, on passe à l'instruction suivante ON I GOSUB 1ØØØ,3ØØØ Si I vaut 1, le sous-programme 1ØØØ est appelé, 2 c'est 3ØØØ.
	ONERR	ONERR GOTO 5ØØ Permet d'intercepter une erreur avant qu'elle ne provoque l'arrêt de l'exécution du programme. Quand une erreur se produit il y a saut à l'instruction indiquée.
	POKE	POKE a,b écrit la donnée b à l'adresse absolue a. (a et b exprimés en décimal) *POKE 2ØØØ,65*
	POP	Désempilement de la dernière adresse de retour (d'un sous-programme)du stack. Le prochain RETURN fera revenir à l'instruction suivant l'avant-dernier GOSUB exécuté.
	PRINT	Affiche un résultat sur écran ou sur imprimante. *PRINT A* *1Ø PRINT A;B;J (juxtaposés)* *3Ø PRINT "X=";X* *2Ø PRINT A,B,J (en zones fixes)*

Mode obligatoire	*Mot-clé*	Définition - *exemples*
	PR#	Transfère la sortie au périphérique dont la carte d'interface est dans le connecteur spécifié par l'argument. *PR#1* permet l'impression si l'interface de l'imprimante est sur le connecteur n°1.
(car.F)	**PR £**	PR £ 3 permet d'activer la MEM et la MEV pour l'affichage en 8Ø colonnes ; faire *'Esc' 'Ctrl/Q'* pour les désactiver.
Programmé	**READ**	Lecture de données dans une instruction DATA associée *1Ø READ A* *2Ø READ B$,C*
Programmé	**REM**	Introduit un commentaire dans le listing du programme.
Direct	**'Ctrl' RESET**	Retour à l'interpréteur ou à l'adresse prévue dans SOFTEV ($3F2,$3F3) si PWERDUP ($3F4) est conforme, sinon le système redémarre comme si on venait de le mettre sous tension (COLDSTART).
	RESTORE	Revient au début des DATA.
	RETURN	Retour de sous-programme *1ØØ RETURN*
	RESUME	Revient à l'instruction d'où était issue l'erreur traitée dans le programme par ONERR GOTO.
	RUN	Déclenche l'exécution d'un programme. Met à zéro toutes les variables. *RUN RUN 3Ø*
	SPEED=	Modifie la cadence d'affichage sur écran de 1 (minimum) à 255 (standard).
	STEP	Introduit le pas d'incrémentation dans FOR.
	STOP	Arrête l'exécution d'un programme *1Ø STOP* et affiche le message *?BREAK IN 1Ø* On peut continuer avec CONT (si les instructions ne sont pas modifiées).

Mode obligatoire	*Mot-clé*	Définition - *exemples*
	TEXT	Affichage en mode texte après un mode graphique. Restaure les valeurs standards de fenêtre d'écran 40 ou 80 caractères par ligne 24 lignes par écran.
	THEN	Introduit l'instruction à effectuer quand un IF est satisfait.
	TO	Introduit la valeur limite d'un FOR.
	TRACE	Met en mode détection et supression éventuelle d'erreurs (debugging). Affiche les numéros d'instructions exécutées sans 'Return' donc entre les lignes de résultats du programme.
	WAIT	Pause conditionnelle dans un programme *WAIT A,B* Suspend l'exécution du programme jusqu'à ce que le contenu de l'adresse A ET (bit à bit) l'équivalent binaire de B soit différent de Ø *WAIT -16384,128* est l'attente qu'une touche soit tapée au clavier.

+	Addition de nombres ou concaténation de chaînes de caractères.
-	Soustraction ou prendre l'opposé.
*	Multiplication.
/	Division.
^	Elévation à la puissance.
=	Egal <>différent.
<	Inférieur >supérieur.
<=	Inférieur ou égal.
=<	Inférieur ou égal.
>=	Supérieur ou égal.
=>	Supérieur ou égal.
NOT	Non logique, agit sur 1 seul opérande Si A est vrai NOT A est faux Si A est faux NOT A est vrai.
AND	ET logique sur 2 opérandes P AND Q n'est vrai que si P <u>et</u> Q sont vrais.
OR	OU logique sur 2 opérandes P OR Q n'est faux que si P <u>et</u> Q sont faux.

] est le caractère de sollicitation du Basic Applesoft de l'Apple II C en clavier QWERTY.

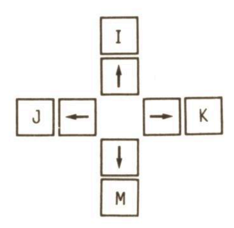
§ est le caractère de sollicitation du Basic Applesoft de l'Apple II C en clavier AZERTY.

Les différentes formes de curseur en Apple //c rappellent si la MEM gérant l'affichage sur 80 colonnes a été activée et si le curseur est en mode déplacement :

▒	Damier carré clignotant	: MEM 80 col. non activée ; 40 col.
▯	Rectangle blanc	: MEM 80 col. active ; 80 col.
⊞	Plus inversé dans rectangle	: 80 col. et curseur en déplacement.
□	Carré blanc	: MEM 80 col. active ; 40 col.
⊞	Plus inversé dans carré	: 40 col. et curseur en déplacement.

- *Déplacement du curseur.*
- *Effacement de lignes sur l'écran.*
- *Recopie de lignes en MEV.*
- *Suppression de lignes de la MEV.*

Les commandes de déplacement du curseur sont obtenues par les huit touches ci-dessous précédées de 'esc'.



En n'appuyant que sur huit touches, on reste en mode déplacement du curseur dans les quatre directions.

Pour revenir au mode normal d'insertion et de correction, on appuie à nouveau sur 'esc'.

Pour effacer à partir de la position du curseur jusqu'en bas de la page, faire 'esc' E.

Pour effacer à partir de la position du curseur jusqu'au bas de la page, faire 'esc' F.

Pour effacer tout l'écran et positionner le curseur en haut à gauche, faire 'esc': @ .

En se servant de la touche ← (BS), on annule le dernier caractère tapé.

En se servant de la touche →, on réenregistre en **MEV** le caractère sous le curseur. Pour réenregistrer une série de caractères, utiliser → en autorépétition, vous recopierez plus vite.

Pour supprimer une ligne d'instructions du programme en mémoire vive, taper le numéro de la ligne et 'Return'.

Les touches 'Del' ou effacement, 'flèche en haut', 'flèche en bas', 'Tab' ne sont opérantes que si le logiciel en cours en tient compte (traitement de texte, par exemple).

En mode d'affichage 8Ø colonnes obtenu par PR£3, la commutation en affichage 4Ø colonnes est obtenue par 'Esc' 4 et le retour en 8Ø colonnes par 'Esc' 8.

Les mots-clés par ordre alphabétique et les codes correspondants en hexadécimal.

Lettres A à S

Mot-clé	Code hexadécimal	Mot-clé	Code hexadécimal	Mot-clé	Code hexadécimal
&	$AF	GR	$88	ON	$B4
ABS	$D4	HCOLOR=	$92	ONERR	$A5
AND	$CD	HGR	$91	OR	$CE
ASC	$E6	HIMEM:	$A3	PDL	$D8
AT	$C5	HLIN	$8E	PEEK	$E2
ATN	$E1	HOME	$97	PLOT	$8D
CALL	$8C	HPLOT	$93	POKE	$B9
CHR$	$E7	HTAB	$96	POP	$A1
CLEAR	$BD	IF	$AD	POS	$D9
COLOR	$A0	IN#	$8B	PRINT	$BA
CONT	$BB	INPUT	$84	PR#	$8A
COS	$DE	INT	$D3	READ	$87
DATA	$83	INVERSE	$9E	RECALL *	$A7
DEF	$B8	LEFT$	$E8	REM	$B2
DEL	$85	LEN	$E3	RESTORE	$AE
DIM	$86	LET	$AA	RESUME	$A6
DRAW	$94	LIST	$BC	RETURN	$B1
END	$80	LOAD *	$B6	RIGHT$	$E9
EXP	$DD	LOG	$DC	RND	$DB
FLASH	$9F	LOMEM:	$A4	ROT =	$98
FN	$C2	MID$	$EA	RUN	$AC
FOR	$81	NEW	$BF	SAVE *	$B7
FRE	$D6	NEXT	$82	SCALE =	$99
GET	$BE	NORMAL	$9D	SCRN(	$D7
GOSUB	$B0	NOT	$C6	SGN	$D2
GOTO	$AB	NOTRACE	$9C	SHLOAD *	$9A
				SIN	$DF

Les mots-clés (suite)

Lettres S à X

Mot-clé	*Code hexadécimal*	*Mot-clé*	*Code hexadécimal*	*Mot-clé*	*Code hexadécimal*
SPC(	$C3	**TAB(**	$CØ	**VAL**	$E5
SPEED=	$A9	**TAN**	$EØ	**VLIN**	$8F
SQR	$DA	**TEXT**	$89	**VTAB**	$A2
STEP	$C7	**THEN**	$C4	**WAIT**	$B5
STOP	$B3	**TO**	$C1		
STORE*	$A8	**TRACE**	$9B	**XPLOT**	
STR$	$E4			**XDRAW**	$95
		USR	$D5		

XPLOT est codé en : $58 $8D
'X' 'PLOT'

Il ne faut pas l'utiliser comme nom de variable.

* mots-clés inutilisables en Apple //c puisque l'interface pour cassettes n'existe plus.

Configuration standard

+ 64K de mémoire vive
+ 2 lecteurs de disquettes

Disquettes du système Pascal

APPLE 1 : } système { Editeur, Filer, Apple, Library
APPLE 2 : } système { Compilateur, Linker, Assembler, etc.
APPLE 3 : Démographiques, Formater, Library

Formatage des disquettes vierges sous PASCAL :

X (exécution) du programme APPLE 3 : FORMATTER
Disquette vierge en D2
Répondre 5 à FORMAT WHICH DISK (4,5,9...12) ? ou 'Return' pour arrêter.

Copier des disquettes :

F (gestion des disquettes) puis **T**(ransfert)
TRANSFER ? Nom de la disquette à recopier :
TO WHERE ? BLANK :
TRANSFER 280 BLOCKS ? **Y**
DESTROY BLANK : ? **Y**

Commandes

E(DIT	Appel de l'éditeur
R(UN	Appel d'un programme source (.TEXT), compilation et exécution
F(ILER	Gestion des fichiers et programmes sur disquettes
C(OMP	Compilation
L(INK	Liaison de programmes déjà compilés
X(ECUTE	Exécution d'un programme objet (.CODE)
A(SSEM	Appel de l'assembleur
D(EBUG	A ne pas utiliser
H	Redémarrage avec une disquette en DOS 3.3

'CTRL A' Visualisation des caractères de la colonne 41 à 80 (partie droite du texte)

'CTRL K' Pour entrer le caractère [

'SHIFT M' Pour entrer le caractère]

>EDIT **A**(DJUST **C**(OPY **D**(ELETE **F**(IND **I**(NSERT **J**(UMP
R(EPLACE **Q**(UIT **X**(CHANGE **Z**(AP **S**(ET **V**(ERIFY

Sens de déplacement dans le texte

>normal avec (→) et (←) ;
<arrière

'CTRL A'	visualise la partie droite
'CTRL L'	déplacement vers le bas
'CTRL O'	déplacement vers le haut
'CTRL Q'	début de ligne
'CTRL Z'	visualisation suit le curseur

Changement de sens

- ou , pour <
+ ou . pour >

<BS> un caractère en arrière avec la flèche gauche (←)
<ETX> fin d'opération par 'CTRL C'
<ESC> annulation d'opération par la touche 'ESC'
<DEL> annulation d'une ligne de texte par 'CTRL X'

Mise en page >**A**DJUST : L(JUST R(JUST C(ENTER <LEFT, RIGHT, UP, DOWN ARROWS> [<ETX> TO LEAVE]

L	décalage du texte sur la marge gauche	
R	justification du texte sur la marge droite	
C	équilibrage du texte au centre	
LEFT	flèche ←	déplacement du texte à gauche
RIGHT	flèche →	déplacement du texte à droite
UP	'CTRL O'	alignement de la ligne précédente
DOWN	'CTRL L'	alignement de la ligne suivante
<EXT>	'CTRL C'	pour sortir

Copie >COPY : B(UFFER F(ROM FILE <ESC>

B copie le tampon (texte venant d'être effacé par exemple)
F copie d'un fichier de la disquette là ou se trouve le curseur
<ESC> 'ESC' pour sortir

Suppression >**D**ELETE : < > MOVING COMMANDS [< EXT > TO DELETE, <ESC> TO ABORT]

	flèche ←	pour effacer un caractère
	flèche →	pour retrouver le texte effacé
EXT	'CTRL C'	pour valider la suppression
ESC	'ESC'	pour annuler la suppression

Vérification >**V**(ERIFY Vérification de l'écran après modifications

Recherche >FIND [1] : L(IT <TARGET> ⇒

Le mot à rechercher est à taper entre / et /. L(ITTERAL si le texte est inclus dans un mot.[n] à la nème occurence du mot. Ce nombre est à indiquer avant de taper F.

Insertion >**I**NSERT TEXT [<BS> A CHAR, <DEL> A LINE] [<ETX> TO ACCEPTS, <ESC> ESCAPES]

Le texte à insérer est tapé normalement avec correction par la flèche ← ou <BS> pour le caractère précédent ou 'CTRL X' pour annuler une ligne. <ETX> ou 'CTRL C' pour valider l'insertion. <ESC> pour annuler l'opération.

Saut >**J**UMP : B(EGINNING E(ND M(ARKER <ESC>

B	pour placer le curseur en début de texte
E	pour placer le curseur en fin de texte
M	pour placer le curseur à des marques préfixées (voir S(ET M(ARKER)
<ESC>	pour annuler

Sortir de l'éditeur >**Q**UIT : **U**(PDATE THE WORFILE AND LEAVE
E(XIT WITHOUT UPDATING
R(ETURN TO THE EDITOR WITHOUT UPDATING
W(RITE TO A FILE NAME AND RETURN

U	en mettant à jour le fichier de travail appelé SYSTEM. WRK.TEXT
E	sortie sans mise à jour
R	retour à l'éditeur sans sauvegarde
W	sauvegarde sur un fichier spécifié et retour à l'éditeur.

Marges >SET : E(NVIRONNEMENT M(ARKER <ESC>
>ENVIRONNEMENT : [OPTIONS] <ETX> OR <SP> TO LEAVE

A(UTO INDENT	TRUE	indentation automatique (alignement sur la ligne précédente)
F(ILLING	FALSE	remplissage jusqu'à la marge droite
L(EFT MARGIN	Ø	marge gauche
R(IGHT MARGIN	78	marge droite
P(ARA MARGIN	5	marge de paragraphe
C(OMMAND CH	^	caractère de commande en mode M(argin)
T(OKEN DEF	TRUE	affecte le mode F(IND et R(EPLACE

En édition d'un programme écrit en langage PASCAL, A doit rester TRUE et F doit être FALSE.

<SP > barre d'espacement pour sortir

Substitution >**R**EPLACE [n] : L(IT V(FY <TARGET> <SUB> ⇨

Remplacement du texte compris entre / et / par un nouveau texte tapé entre / et / de longueur quelconque.

L pour remplacer une partie d'un mot

n nombre d'opérations de substitution (à donner avant d'appeler R)

Echange de caractères >E**X**CHANGE : TEXT [<BS> A CHAR]
[<ESC> ESCAPES, <ETX> ACCEPTS]

Le caractère remplace celui sous le curseur

<BS> flèche ← pour le caractère précédent

<ESC> 'ESC' pour annuler

<ETX> 'CTRL C' pour valider

Effacement >**Z**AP effacement depuis la position courante jusqu'à celle de début du dernier texte trouvé, remplacé ou inséré.

FILER **G**(ET **S**(AVE **N**(EW **L**(IST DIRECTORY **E**(XTENDED-DIRECTORY LIST **R**(EMOVE **C**(HANGE **T**(RANSFER **D**(ATE **Q**(UIT **V**(OLUME **W**(HAT **B**(AD-BLOCKS **X**(AMINE **Z**(ERO **P**(REFIX

G GET ? **Nom d'une disquette : Nom du programme**
Chargement en mémoire du programme désigné, il remplace le SYSTEM.WRK.TEXT (fichier de travail)
TEXT FILE LOADED, l'opération est réalisée.

S SAVE AS ? **Nom d'une disquette : Nom du programme**
Sauvegarde du fichier de travail sous le nom spécifié dans la disquette désignée.
TEXT FILE SAVED, l'opération est réalisée.

N Effacement du fichier de travail en MEV et sur disquette
Réponse : WORKFILE CLEARED.

L DIR LISTING OF ? **Nom de la disquette :**
Affiche le contenu de la disquette. (catalogue)
Faire suivre le nom par ,PRINTER : pour imprimer sur papier.

E Affichage du contenu avec des informations diverses comme les zones inutilisées.
DIR LISTING OF ? **Nom de la disquette :**

R Suppression d'un fichier.

C Changement de nom d'un fichier ou d'une disquette.

T Transfert d'une disquette ou d'un fichier sur une autre disquette.
TRANSFER ? **Nom de la disquette : [Nom du programme]**
TO WHERE ? **Nom de la disquette : [Nom du programme]**
Pour imprimer un programme source répondre PRINTER: à la question TO WHERE ?

D Mise à jour de la date.
Jour - Mois (3 lettres) - Année (2 chiffres)

Q Sortir du Filer.

V Liste des volumes connus du système par leur numéro et leur nom.

W WHAT donne le nom du fichier de travail et indique s'il a été sauvegardé ou non.

B Bad-blocks teste les 280 blocs d'une disquette et signale les blocs en mauvais état physique.

X Examine les mauvais blocs et essaie de les rendre cohérents (les répare). Si ce n'est pas possible il permet le marquage des mauvais blocs.
(Opération utile avant l'utilisation d'une disquette vierge).

Z Zéro efface le DIRECTORY (liste des fichiers).

P Préfix permet le changement de nom du volume courant pris par défaut (si on tape seulement :) par le volume spécifié.

COMMANDES MONITOR

Le signe indicatif est le caractère *

Les données sont fournies en numération hexadécimale.

Les adresses sont données sous forme de 4 chiffres hexadécimaux ou moins.

Commande	Définition - *exemples*
Adresse G	Exécution du programme commençant à cette adresse. *3DØ G : redémarrage du Basic à chaud*
Adresse L	Listing des 20 instructions en langage machine à partir de cette adresse ; désassemblage des codes hexadécimaux en code mnémonique du mini-assembleur. *3DØ L* *3DØ - 4C BF 9D JMP $9DBF* *3D3 - 4C 84 9D JMP $9D84* *...etc*
Adresse 1. adresse 2	Affichage des contenus des positions de mémoire à partir de l'adresse 1 jusqu'à l'adresse 2. *3DØ.3D7* *3DØ - 4C BF 9D 4C 84 9D 4C FD* Les adresses de début de ligne sont toujours de la forme XXXØ ou XXX8 sauf éventuellement l'adresse 1.
Adresse	Si une seule adresse est spécifiée, le Monitor renvoie le contenu de cette position de mémoire. *3D1* *3D1 - BF*
Adresse : valeur 'esp' valeur	Modification ou écriture de valeurs dans des positions de mémoire adjacentes. *73: ØØ 2Ø (modification de HIMEM)*

Commande	Définition - *exemples*
Adresse 1 < adresse 2. adresse 3 M	Déplacement d'une zone de valeurs contenues dans adresse 2 à adresse 3, dans la zone commençant en adresse 1. *6ØØØ <4ØØ.7FF M* sauvegarde de la page écran texte ou graphique entre $6ØØØ et $63FF
Adresse 1 <adresse 2. adresse 3 V	Vérification de l'identité de 2 zones de mémoire. Dès qu'une différence apparaît elle est signalée : *F1ØB < B1.C8 V* *ØØB8 - Ø5 (6Ø)* *ØØB9 - Ø2 (EA)* Le sous-programme CHRGET sous sa forme d'origine diffère de la forme de travail par le contenu de $B8 et $B9 (pointeur dans le texte BASIC).
N	Affichage en mode normal et séparateur de commandes successives au MONITOR.
I	Affiche en mode inverse (noir sur blanc).
Valeur ± valeur	Opérations d'addition et de soustraction dans le système de numération hexadécimale (à 2 chiffres). *3F+Ø1 ou 4Ø-Ø1* *4Ø 3F*
Numéro de connecteur 'Ctrl K'	Transfert sur une entrée issue du périphérique branché sur le connecteur spécifié. Transfert du contrôle des sorties au périphérique dont la carte d'interface est installée sur le connecteur spécifié. *6 'ctrl P'* stimule la PROM de la carte contrôleur du lecteur de disquette et provoque l'amorçage du chargement du SED en mémoire vive depuis une disquette.
'Ctrl B'	Amorçage de l'interpréteur BASIC en MEM (démarrage à froid).

Commande	Définition - *exemples*
'Ctrl C'	Re-amorçage de l'interpréteur BASIC en MEM (idem à 'Reset').
'Ctrl E'	Affichage du contenu des registres du microprocesseur. *A= X= Y= P= S=* Modification de ces registres
'Ctrl Y'	Branchement au programme qui commence à l'adresse. *$3F8* *3F8 : 4C ØØ Ø3 JMP $3ØØ* $3F8 doit être préparée pour provoquer un branchement vers le début du programme appelé.

Sauvegarde - chargement de programmes

LOAD NOM,D1	Charge en MEV le programme NOM depuis la disquette placée dans le lecteur 1.
SAVE NOM,D2	Sauve le programme en BASIC de la MEV sur la disquette placée dans le lecteur 2.
BLOAD BINAIRE	Charge en MEV le fichier binaire BINAIRE depuis la disquette, à l'adresse absolue indiquée dans l'en-tête du fichier. Cette adresse et la longueur sont en MEV à $AA72 et $AA60 après le chargement.
BSAVE BINAIRE, A$300,L$7F	sauve le programme en langage machine implanté à partir de l'adresse absolue $300 sur une longueur de $7F octets, sur la disquette actuelle sous le nom BINAIRE.
BSAVE IMAGE,A$2000,L$1FF8	sauve l'image graphique haute résolution page 1, sur la disquette sous le nom IMAGE.
HGR **BLOAD IMAGE**	Restitue sur l'écran graphique haute résolution, les points enregistrés dans IMAGE.
RUN NOM	Charge en MEV et exécute le programme intitulé NOM sur la disquette.
BRUN BINAIRE	Charge en MEV et exécute le programme en langage BINAIRE enregistré sur la disquette.
CHAIN ENTIER	Charge en MEV le programme écrit en Basic INTEGER sans effacer la zone des variables du programme précédent. Le programme ENTIER ne doit pas redimensionner les variables communes.

D$=CHR$(13)+CHR$(4)
Fichiers séquentiels (type T)

PRINT D$"OPEN TS"	Ouvre le fichier dénommé TS sur la disquette actuellement en ligne. Place le pointeur au début du fichier séquentiel.
PRINT D$"OPEN"F$",D1"	Ouvre un fichier de nom variable F$ sur la disquette placée sur le lecteur n° 1.
PRINT D$"READ TS"	Prépare une opération de lecture au début du fichier séquentiel TS, préalablement ouvert par OPEN.

INPUT A$	Lit dans le fichier TS une chaîne de caractères qui sera mémorisée en MEV sous le nom A$. Le pointeur est déplacé au début du champ de données suivant.
GET C$	Lit un seul caractère et déplace le pointeur d'un caractère.
PRINT D$"POSITION TS,R"P	positionne le pointeur après le Pème 'Return' depuis la position actuelle.
PRINT D$"WRITE TS"	Prépare une opération d'écriture sur le fichier TS là où se trouve le pointeur.
PRINT X$	Ecrit dans le fichier TS la chaîne X$.
PRINT Y$	Ecrit la chaîne Y$ séparée de la chaîne X$, précédemment enregistrée, par le caractère 'Return'.
PRINT CHR$(4)"CLOSE TS"	ferme le fichier TS en sauvant sur disquette le tampon de sortie alloué à ce fichier par OPEN, contenant le dernier secteur utilisé.
PRINT D$"APPEND TS"	Réouvre le fichier TS en positionnant le pointeur à la fin du fichier. Permet d'écrire des données en les rajoutant à la fin du fichier.
PRINT D$"READ TS,B17"	Positionne le pointeur à l'octet n° 17 pour une future lecture. (Ø est le 1er octet).

D$=CHR$(13)+CHR$(4)
Fichiers à accès direct (type T)

PRINT D$"OPEN TACT,L21"	ouvre le fichier TACT sur une disquette en prévoyant des enregistrements de longueur constante ; ici 21 octets y compris le 'Return' pris comme fin d'enregistrement.
PRINT D$"OPEN"F$",L"N",D2"	ouvre un fichier de nom variable F$, de longueur égale à la variable N, sur la disquette placée dans le lecteur n° 2.
PRINT D$"READ TACT,R"I	positionne le pointeur sur le début du Ième enregistrement pour une future lecture.
INPUT A$	Récupère dans A$, le contenu du Ième enregistrement du fichier TACT.
GET C$	Lit le caractère situé sous le pointeur dans l'enregistrement n° I.

PRINT D$"WRITE TACT,R"J Positionne le pointeur sur le début du Jème enregistrement pour une future écriture.

**PRINT X$** Ecrit la chaîne X$ dans l'enregistrement n° J.

**PRINT Y$** Ecrit la chaîne Y$, séparée de la chaîne précédemment enregistrée, par 'Return'.

PRINT CHR$(4)"CLOSE TACT" ferme le fichier TACT en sauvegardant le contenu du tampon de sortie alloué à ce fichier.

PRINT D$"READ TACT,RØ,B"K positionne le pointeur à l'octet n° K de l'enregistrement n° Ø, pour une lecture ultérieure.

PRINT CHR$(4) Annule l'effet d'une commande du SED, comme READ, par exemple pour donner à INPUT le sens d'une entrée par le clavier.

Commandes diverses

CATALOG D2 Affiche la liste des programmes et des fichiers enregistrés sur la disquette placée en D2.

* fichier verrouillé

I, A, T, B types de fichiers :

- I : Basic Integer
- A : Basic Applesoft
- T : Fichiers T séquentiels ou directs
- B : Binaires, données ou sous-programmes en langage machine

ØØ2 : nombre de secteurs occupés par le fichier (modulo 256)

Un secteur comprend 256 octets utiles.
Une piste 16 secteurs.
Une disquette comprend 35 pistes dont 31 sont utilisables et 4 sont réservées au SED

Le nombre maximum de références dans le répertoire d'une disquette de 105 en DOS 3.3.

MON C,I,O

Affiche les commandes, les entrées et les sorties telles qu'elles sont reçues ou envoyées par le SED pendant l'exécution d'un programme.
Ce mode est annulé par 'Reset'.

NOMON C,I,O

Annule le mode précédent.

PRINT D$"PR £"S

Met le périphérique branché sur le connecteur n° S en ligne pour l'ordre PRINT à suivre.
Si l'imprimante à sa carte d'interface dans le connecteur n° 1 :
PRINT D$"PR # 1"

PRINT D$"PR £Ø"

Désactive le périphérique de sortie et seul l'écran reste en ligne.

PRINT D$"IN £"S

Connecte le périphérique branché sur le connecteur S, pour qu'il puisse envoyer des données au système.

PRINT D$"IN £Ø"

L'entrée des données est limitée au clavier.

MAXFILES 4

Prévoit d'utiliser 4 tampons d'entrée/sortie en parallèle correspondant à 4 fichiers ouverts disponibles en MEV.
Chaque tampon occupe 595 octets.
Par défaut le système réserve 3 zones tampons.
Cette commande doit être exécutée avant de charger et d'exécuter un programme.

VERIFY NOM

Teste le bon enregistrement physique du programme ou du fichier NOM. Si un secteur de la disquette utilisée n'est pas bon à l'écriture, le message I/O ERROR sera affiché (est réalisée automatiquement après SAVE).

Manipulation de fichiers

INIT HELLO [,V254]

Initialisation d'une disquette vierge.
Le programme Basic en MEV est chargé sur la disquette sous le nom HELLO, ainsi que le SED.
La disquette porte un numéro de volume qui peut servir de contrôle.
Le formatage de la disquette est le suivant :

16 secteurs par piste. La place disponible est 126976 octets par disquette.

Le système d'exploitation enregistré sur la disquette est de type esclave, il ne contient plus de programme de relocation pour s'adapter au mieux à la configuration-mémoire.

Cette disquette maintenant initialisée est suffisante pour démarrer le système.

DELETE NOM — Supprime le fichier NOM de la disquette sauf s'il est LOCKé (verrouillé).

RENAME ANCIEN, NOUVEAU — Change le nom d'un fichier ou d'un programme.

LOCK NOM — Verrouille le fichier NOM contre toute tentative de suppression par DELETE ou de réenregistrement par SAVE ou WRITE.

UNLOCK NOM — Déverrouille le fichier NOM.

Fichier de commandes

```
LIST
  10 PRINT D$"OPEN CMD"          Création du fichier
  20 PRINT D$"WRITE CMD"
  30 PRINT "PR £1"               Enregistrement des ordres
  40 PRINT "CHR$(9)"60 N"
  50 PRINT "LIST"
  60 PRINT "PR £0"
  70 PRINT D$"CLOSE CMD"         Fermeture du fichier
RUN                              Réalisation du fichier
```

EXEC CMD — Exécution automatique des commandes enregistrées dans CMD c'est-à-dire dans l'exemple :
impression d'un listing sur imprimante à raison de 60 caractères par ligne.

Toutes les instructions et commandes exécutables en mode direct sont susceptibles d'appartenir à un fichier de ce type.

Changement d'interpréteur Basic

INT	Mise en opération de l'interpréteur Integer Basic en MEV à banc-commuté de l'Apple //c. Le curseur succède à >
FP	Mise en opération de l'interpréteur Basic-Applesoft (en MEM carte-mère). Le curseur apparaît à côté de] ou §.

Une disquette formatée en **PRODOS** contient des fichiers de divers types et plus particulièrement le fichier-répertoire de la disquette dont le nom est associé à tout le contenu de la disquette. A chaque disquette ou volume correspond donc un nom unique qui est le nom du fichier-répertoire de cette disquette.

Affichage du répertoire d'une disquette

CATALOG — Affiche sur 8Ø colonnes la liste des fichiers, chaque fichier étant caractérisé par huit paramètres de gauche à droite.
* un astérisque si le fichier est verrouillé :
. nom du fichier ;
. type de fichier (SYS,BAS,TXT,DIR,VAR, REL,$Fn) ;
. nombre de blocs de 512 octets occupés ;
. date à laquelle le fichier a été modifié ;
. date à laquelle le fichier a été créé ;
. fin logique du fichier, c'est-à-dire le nombre théorique d'octets utilisés si tous les enregistrements du fichier étaient occupés ;
. adresse de chargement d'un fichier binaire ou bien
. longueur des enregistrements d'un fichier à accès direct.
Sous la liste, apparaît le nombre de blocs libres, de blocs utilisés et le nombre total de blocs.

CAT — Affiche sur 4Ø colonnes la liste des fichiers du répertoire, chaque fichier étant décrit par les cinq premières rubriques précédentes.

Le nombre de fichiers par répertoire est limité à 51 fichiers.

Création d'un fichier-sous-répertoire

CREATE /RP/SR — Création du fichier SR de type DIR ou répertoire sur la disquette de répertoire principal /RP/.

CAT /RP/SR — Affiche la liste des fichiers répertoriés dans SR.

Affichage ou affectation d'un préfixe

PREFIX	Affiche la valeur du préfixe (généralement le nom attribué à un volume, c'est-à-dire une disquette).
PREFIX /EXEMPLES	Affecte la valeur /EXEMPLES/ au préfixe.
PREFIX,D1	Affecte au préfixe le nom de la disquette se trouvant dans le lecteur n° 1.
PREFIX /	Donne la valeur vide au préfixe.

Dans un programme, cette commande suivie d'une instruction INPUT fournit la valeur du préfixe.

Exemple

```
 5 D$=CHR$(4)
10 PRINT D$ "PREFIX"
20 INPUT PFIX$
30 PRINT "Cette disquette s'appelle";PFIX$
RUN
```

Cette disquette s'appelle /EXEMPLES/.

Le préfixe est aussi le nom du répertoire principal de la disquette.

Identification d'un fichier

Un nom de fichier comporte au maximum 15 caractères (lettres, chiffres et point seulement) et débute par une lettre ; les lettres minuscules sont automatiquement converties en majuscules.

L'accès à un fichier se fait par un "chemin" commençant par le préfixe suivi du nom du fichier. Le "chemin" d'accès à un fichier ne doit pas comporter plus de 64 caractères.

Exemple de "chemin"

/EXEMPLES/NOMFICHIER

Un chemin est dit "partiel" s'il ne commence pas par /. PRODOS mettra le préfixe en tête de ce chemin partiel à moins que ce préfixe soit vide ou bien que la commande spécifie le numéro du lecteur et du connecteur (par exemple : NOMFICHIER, D2, S6).

Sauvegarde-chargement de fichiers

LOAD /EXEMPLES/NOM	Charge en MEV le programme NOM de la disquette /EXEMPLES/. Efface tout autre programme en Basic de la MEV.

SAVE /EXEMPLES/NOM — Sauve le programme écrit en Basic de la MEV sur la disquette dénommée /EXEMPLES/, sous l'intitulé NOM. Tout autre programme du même nom sera écrasé par le dernier sauvé sur disquette.

BLOAD /EXEMPLES/BINAIRE — Charge en MEV le fichier BINAIRE depuis la disquette /EXEMPLES/. L'adresse absolue du chargement est dans l'en-tête du fichier. Cette adresse peut être forcée à une autre valeur en adjoignant à la commande l'option **,A$a**. Cette commande s'applique à tous les types de fichiers, elle charge l'image binaire de ce fichier à partir de l'adresse $a. La longueur ou le nombre d'octets à charger est donnée dans l'option ,L$1 ou bien par l'adresse du dernier octet à charger par l'option ,E$d.

BSAVE /EXEMPLES/BINAIRE, A$300, L$7F — Sauve le programme en langage machine implanté à partir de l'adresse $300 sur une longueur $7F octets, sur la disquette /EXEMPLES/.

BSAVE /EXEMPLES/IMAGE, A$2000, L$1FFF — Sauve l'image graphique haute résolution page 1 (8K octets) sur la disquette /EXEMPLES/ sous le nom IMAGE. L'option ,E$3FFF peut remplacer ,1$1FFF.

HGR
BLOAD IMAGE — Restitue sur l'écran graphique les points enregistrés dans IMAGE sur la disquette.

CHAIN /EXEMPLES/DEUX — Charge et fait exécuter le programme DEUX sans effacer la zone des variables du programme précédent pour pouvoir se servir des données et résultats du premier.

STORE /EXEMPLES/VARB — Rassemble toutes les variables définies dans le programme en Basic et les sauvent dans le fichier VARB de type VAR. Les chaînes de caractères seront compactées avant la sauvegarde.

RESTORE /EXEMPLES/VARB — Efface toutes les variables du programme en Basic et les remplace par celles du fichier VARB dans la zone correspondante en MEV.

FRE — Efface toutes les chaînes restantes d'un précédent programme.

Exécution d'un programme

RUN /EXEMPLES/NOM	Charge en MEV et lance l'exécution du programme NOM de la disquette /EXEMPLES/.
BRUN /EXEMPLES/BINAIRE	Charge en MEV et lance l'exécution du programme BINAIRE.
-/EXEMPLES/X	La commande - (DASH ou tiret) est l'abréviation de RUN ou de BRUN et aussi EXEC. Elle s'applique aux fichiers de type BAS, BIN, TXT et SYS.

Fichiers de type TXT

Huit peuvent être gérés simultanément en MEV.

D$=CHR$(4)
Fichier séquentiel

PRINT D$ "OPEN /E/FS"	Ouvre le fichier séquentiel FS. Une zone tampon de 1Ko en MEV est allouée et le pointeur est au début du fichier. Si le fichier n'existe pas encore, il est créé.
PRINT D$ "READ /E/FS"	Prépare une opération de lecture sur le fichier FS à la position donnée par le pointeur.
PRINT D$ "READ /E/FS,F"N	Positionne le pointeur après le Nème 'Return' du fichier séquentiel. Le 'Return' signale la fin d'un champ (Field).
INPUT A$	Lit dans le fichier FS une chaîne de caractères qui sera mémorisée en MEV sous le nom A$. Le pointeur sera déplacé au début du champ suivant.
GET C$	Lit un seul caractère et déplace le pointeur d'un seul caractère.
PRINT D$ "POSITION /E/FS,F"N	Positionne le pointeur après le Nème 'Return' depuis la position actuelle.
PRINT D$ "WRITE /E/FS"	Prépare une opération de lecture à la position donnée par le pointeur.
PRINT D$ "WRITE /E/FS,F"N	Positionne le pointeur après N champs pour la prochaine écriture.

PRINT X$	Ecrit dans le fichier FS la valeur de X$ suivie d'un 'Return'.
PRINT Y$	Ecrit à la suite dans le fichier FS la valeur de Y$ suivie de 'Return'.
PRINT D$ "CLOSE /E/FS"	Ferme le fichier FS en sauvant sur disquette la zone-tampon de sortie allouée à ce fichier et libère cette zone pour d'autres fichiers.
PRINT D$ "APPEND /E/FS"	Réouvre le fichier FS en positionnant le pointeur en fin de fichier et prépare à l'écriture de données à la suite du fichier.
PRINT D$ "FLUSH /E/FS"	Normalement, les données sont transférées sur disquette par blocs de 512 octets, mais avec cette commande, elles sont transférées immédiatement à chaque PRINT.

Fichiers à accès direct

PRINT D$ "OPEN /E/FAD,L"LE	Ouvre un fichier à accès direct FAD en prévoyant une longueur fixe des enregistrements égale à LE, y compris le caractère 'Return' pris comme fin d'enregistrement. Si la longueur n'est pas spécifiée, elle est égale à celle spécifiée à la création du fichier.
PRINT D$ "READ /E/FAD,R"I	Positionne le pointeur au début du Ième enregistrement du fichier FAD à accès direct pour une future lecture.
INPUT A$	Récupère dans A$ le contenu du premier champ du Ième enregistrement du fichier FAD.
GET C$	Lit le caractère situé sous le pointeur dans l'enregistrement n° 1.
PRINT D$ "WRITE /E/FAD,R"J	Positionne le pointeur au début du Jème enregistrement pour une future opération d'écriture.
PRINT X$	Ecrit la chaîne X$ dans l'enregistrement n° 1 suivi du caractère 'Return'.
PRINT Y$	Ecrit le contenu de la chaîne Y£ à la suite suivi d'un 'Return'.

PRINT D$ "READ /E/FAD/,RØ,B"K	Positionne le pointeur à l'octet n° K de l'enregistrement n° Ø pour une lecture.
PRINT D$ "CLOSE /E/FAD"	Ferme le fichier FAD en sauvegardant les dernières données de la zone-tampon de sortie allouée à ce fichier et libère cette zone pour d'autres fichiers.
PRINT D$ "CLOSE"	Ferme tous les fichiers ouverts.

Manipulation de fichiers

DELETE /EXEMPLES/NOM	Supprime le fichier NOM du répertoire /EXEMPLES/ de la disquette /EXEMPLES/ sauf s'il est verrouillé (*).
LOCK /EXEMPLES/NOM	Verrouille le fichier NOM contre toute tentative de suppression, de sauvegarde ou d'écriture sur ce fichier, de changement de nom.
UNLOCK /EXEMPLES/NOM	Déverrouille le fichier NOM.
RENAME ANCIEN,NOUVEAU	Change le nom d'un fichier qui doit rester dans le même répertoire.

Formatage d'une disquette vierge sous PRODOS

- **Démarrer** le système avec la disquette /UTILITAIRES/.
- **Choisir** l'option 6-Formater Volume puis :
- **Indiquer** sur quel lecteur se trouve la disquette à formater.
- **Introduire** la disquette dans le lecteur choisi.
- **Choisir** l'option 1-PRODOS pour formater la disquette avec PRODOS.
- **Entrer** le nom du nouveau volume : /EXEMPLES/.

Le formatage s'effectuera.

Préparation d'une disquette de travail en Basic

Une fois formatée, la disquette peut recevoir des fichiers. Les trois fichiers suivants sont indispensables :

* PRODOS
* BASIC.SYSTEM
* STARTUP

Pour les deux premiers, Il suffit de réutiliser la disquette /UTILITAIRES/ et de choisir l'option 1-Copier fichier, ensuite d'indiquer les lecteurs source et destination de la copie, puis de sélectionner ces fichiers présents dans la disquette /UTILITAIRES/ et de les faire recopier sur la nouvelle.

Le fichier **STARTUP** est généralement un programme écrit en Basic ; en l'appelant STARTUP, il sera le programme mis en exécution par le PRODOS à la mise sous tention après les différentes phases de chargement du système en MEV.

Par exemple:

```
NEW
1Ø HOME
2Ø D$=CHR$(4)
3Ø PRINT D$ "CAT"
SAVE STARTUP
```

créé un programme STARTUP affichant automatiquement la liste des fichiers de la disquette.

Le type de fichier dénommé STARTUP est libre (Binaire, Basic ou Exec).

Fichiers de commandes (type TXT)

La création d'un fichier de commandes est identique à celle d'un fichier séquentiel où les données sont des commandes exécutables immédiatement par le système.

EXEC /EXEMPLES/CMD Exécute automatiquement les commandes lues dans le fichier CMD.

Fichiers en MEV auxiliaire

Le nom de Volume /RAM permet l'exploitation de fichiers-programmes ou données implantés en MEV. (Le temps d'accès est bien sûr très supérieur). L'espace en mémoire auxiliaire est de 61 K octets ou 128 blocs, mais 8 servent au système pour le repérage.

La commande CAT/RAM affiche la liste des fichiers de cet espace et le nombre de blocs disponibles.

Commandes à l'imprimante connectée sur le PORT1

Après avoir mis l'imprimante en ligne par PR£1 :

- faire précéder toutes les commandes de PRINT CHR$(9) ou bien PRINT"'Ctrl I'" ;
- le caractère de sollicitation de commandes apparaît sous la forme d'un point d'interrogation clignotant.

<table>
<tr><th>Commandes</th><th>Rôle</th></tr>
<tr><td>nnnN</td><td>Définit la largeur d'impression en un nombre de caractères compris entre 1 et 255 (8Ø au départ).</td></tr>
<tr><td>nnB</td><td>Définit le débit en un nombre correspondant aux valeurs suivantes en BAUD :
<table>
<tr><th>nn</th><th>Baud</th><th>nn</th><th>Baud</th></tr>
<tr><td>1</td><td>5Ø</td><td>9</td><td>18ØØ</td></tr>
<tr><td>2</td><td>75</td><td>1Ø</td><td>24ØØ</td></tr>
<tr><td>3</td><td>11Ø</td><td>11</td><td>36ØØ</td></tr>
<tr><td>4</td><td>135</td><td>12</td><td>48ØØ</td></tr>
<tr><td>5</td><td>15Ø</td><td>13</td><td>72ØØ</td></tr>
<tr><td>6</td><td>3ØØ</td><td>* 14</td><td>96ØØ</td></tr>
<tr><td>7</td><td>6ØØ</td><td>15</td><td>192ØØ</td></tr>
<tr><td>8</td><td>12ØØ</td><td></td><td></td></tr>
</table></td></tr>
<tr><td>nD</td><td>Définit le format des données d'après n :
<table>
<tr><th>n</th><th>Bits de données</th><th>Bits Stop</th></tr>
<tr><td>Ø</td><td>8</td><td>1</td></tr>
<tr><td>1</td><td>7</td><td>1</td></tr>
<tr><td>2</td><td>6</td><td>1</td></tr>
<tr><td>3</td><td>5</td><td>1</td></tr>
<tr><td>4</td><td>8</td><td>2</td></tr>
<tr><td>5</td><td>7</td><td>2</td></tr>
<tr><td>6</td><td>6</td><td>2</td></tr>
<tr><td>* 7</td><td>~~8~~ 5</td><td>2</td></tr>
</table></td></tr>
<tr><td>I</td><td>Affiche les caractères sur l'écran pendant l'impression.</td></tr>
<tr><td>K</td><td>Empêche le saut à la ligne automatique après 'Return'.</td></tr>
<tr><td>L</td><td>* Envoie un saut à la ligne après chaque 'Return'.</td></tr>
<tr><td>nP</td><td>Définit la parité : Ø, 2, 4, 6 : pas de parité.
n=1 : parité impaire ; n=3 : parité paire ;
n=5 : MARK(1) ; n=6 : SPACE(Ø).</td></tr>
<tr><td>R</td><td>Réinitialise les paramètres et désactive les S/P.</td></tr>
<tr><td>S</td><td>Envoie un caractère BREAK de 233 millisecondes.</td></tr>
<tr><td>Z</td><td>Ignore les prochains caractères de commande jusqu'à ce qu'on tape Ctrl/Reset ou PR£1.</td></tr>
</table>

- les astérisques indiquent les valeurs données par le système au départ.

Commandes au MODEM connecté au PORT2

Les ordres suivants mettent le MODEM en ligne :

IN£2 : assigne au PORT2 l'entrée de données (mode terminal).
PR£1 : renvoie sur l'imprimante les données entrées.
PR£2 : assigne au PORT2 la sortie des données.

Puis les commandes doivent être précédées de PRINT CHR$(1) ou PRINT"'CtrlA'".

Le caractère de sollicitation des commandes apparaîtra en point d'interrogation clignotant.

Commandes	*Rôle*
nnnN **nnB** **nP**	Voir la table des commandes à l'imprimante du PORT1.
T	Met en mode terminal. S'utilise après IN£2. Le caractère de sollicitation devient _ clignotant.
Q	Quitte le mode terminal.
R	Réinitialise les paramètres et désactive les S/P.
S	Envoie un caractère BREAK de 233 millisecondes.
Z	Ignore les prochains caractères de commande.
Control T	Depuis un système à distance, met l'Apple II C en mode terminal (s'il avait déjà reçu IN£2).
Control R	Depuis un ordinateur à distance, fait quitter le mode terminal à l'Apple II C.

Valeurs données au départ du système :

- 3ØØ baud ;
- 8 bits de données, 1 bit Stop, pas de bit de parité ;
- pas de saut à la ligne après 'Return' ;
- pas de 'Return' dans le flot de sortie ;
- pas d'affichage sur l'écran.

Il est recommandé de se servir de la disquette "Utilitaires Système" pour redéfinir les configurations des Ports série 1 et 2.

CONVERSION HEXADECIMALE/DECIMALE/HEXADECIMALE
des 256 premiers nombres $00 à $FF

$00	:0	1	2	3	4	5	6	7
$08	:8	9	10	11	12	13	14	15
$10	:16	17	18	19	20	21	22	23
$18	:24	25	26	27	28	29	30	31
$20	:32	33	34	35	36	37	38	39
$28	:40	41	42	43	44	45	46	47
$30	:48	49	50	51	52	53	54	55
$38	:56	57	58	59	60	61	62	63
$40	:64	65	66	67	68	69	70	71
$48	:72	73	74	75	76	77	78	79
$50	:80	81	82	83	84	85	86	87
$58	:88	89	90	91	92	93	94	95
$60	:96	97	98	99	100	101	102	103
$68	:104	105	106	107	108	109	110	111
$70	:112	113	114	115	116	117	118	119
$78	:120	121	122	123	124	125	126	127
$80	:128	129	130	131	132	133	134	135
$88	:136	137	138	139	140	141	142	143
$90	:144	145	146	147	148	149	150	151
$98	:152	153	154	155	156	157	158	159
$A0	:160	161	162	163	164	165	166	167
$A8	:168	169	170	171	172	173	174	175
$B0	:176	177	178	179	180	181	182	183
$B8	:184	185	186	187	188	189	190	191
$C0	:192	193	194	195	196	197	198	199
$C8	:200	201	202	203	204	205	206	207
$D0	:208	209	210	211	212	213	214	215
$D8	:216	217	218	219	220	221	222	223
$E0	:224	225	226	227	228	229	230	231
$E8	:232	233	234	235	236	237	238	239
$F0	:240	241	242	243	244	245	246	247
$F8	:248	249	250	251	252	253	254	255

LES CARACTERES ALPHANUMERIQUES ET LEURS CODES ASCII ET ECRAN

Dec	*Hex*	*Caractère Clavier*	*Nom*	*Ecran*	*Dec*	*Hex*	*Caractère Clavier*	*Ecran*
ØØ	ØØ	c à	NUL	à i	32	2Ø	Espace	Espace i
Ø1	Ø1	c A	SOH	A i	33	21	!	! i
Ø2	Ø2	c B	STX	B i	34	22	"	" i
Ø3	Ø3	c C	ETX	C i	35	23	£	£ i
Ø4	Ø4	c D	EOT	D i	36	24	$	$ i
Ø5	Ø5	c E	ENQ	E i	37	25	%	% i
Ø6	Ø6	c F	ACK	F i	38	26	&	& i
Ø7	Ø7	c G	BEL	G i	39	27	'	' i
Ø8	Ø8	c H ←	BS	H i	4Ø	28	(	(i
Ø9	Ø9	c I →|	HT	I i	41	29	)	) i
1Ø	ØA	c J ↓	LF	J i	42	2A	*	* i
11	ØB	c K ↑	VT	K i	43	2B	+	+ i
12	ØC	c L	FF	L i	44	2C	, virgule	, i
13	ØD	c M	CR	M i	45	2D	- tiret	- i
14	ØE	c N	SO	N i	46	2E	. point	. i
15	ØF	c O	SI	O i	47	2F	/ division	/ i
16	1Ø	c P	DLE	P i	48	3Ø	Ø	Ø i
17	11	c Q	DC1	Q i	49	31	1	1 i
18	12	c R	DC2	R i	5Ø	32	2	2 i
19	13	c R	DC3	S i	51	33	3	3 i
2Ø	14	c T	DC4	T i	52	34	4	4 i
21	15	c U →	NAK	U i	53	35	5	5 i
22	16	c V	SYN	V i	54	36	6	6 i
23	17	c W	ETB	W i	55	37	7	7 i
24	18	c X	CAN	X i	56	38	8	8 i
25	19	c Y	EM	Y i	57	39	9	9 i
26	1A	c Z	SUB	Z i	58	3A	: deux pts	: i
27	1B	Esc	ESC	° i	59	3B	; pt.virg.	; i
28	1C	c ç	FS	ç i	6Ø	3C	<	< i
29	1D	c §	GS	§ i	61	3D	= égal	= i
3Ø	1E	c ^	RS	^ i	62	3E	>	> i
31	1F	c _	US	_ i	63	3F	?	? i

c X pour Ctrl X

i pour affichage INVERSE

Icones : se référer au programme "Comment" n° 17 "Faire afficher les icones".

Dec	Hex	Caractère-écran-			Dec	Hex	Caractère-écran-		
		Clavier	Prim	Alt *			Clavier (//c)	Prim	Alt
64	4Ø	à	à cl	à i	96	6Ø	`	esp. cl	` i
65	41	A	A cl	A i	97	61	a	! cl	a i
66	42	B	B cl	B i	98	62	b	" cl	b i
67	43	C	C cl	C i	99	63	c	£ cl	c i
68	44	D	D cl	D i	1ØØ	64	d	$ cl	d i
69	45	E	E cl	E i	1Ø1	65	e	% cl	e i
7Ø	46	F	F cl	F i	1Ø2	66	f	& cl	f i
71	47	G	G cl	G i	1Ø3	67	g	' cl	g i
72	48	H	H cl	H i	1Ø4	68	h	(cl	h i
73	49	I	I cl	I i	1Ø5	69	i	) cl	i i
74	4A	J	J cl	J i	1Ø6	6A	j	* cl	j i
75	4B	K	K cl	K i	1Ø7	6B	k	+ cl	k i
76	4C	L	L cl	L i	1Ø8	6C	l	, cl	l i
77	4D	M	M cl	M i	1Ø9	6D	m	- cl	m i
78	4E	N	N cl	N i	11Ø	6E	n	. cl	n i
79	4F	O	O cl	O i	111	6F	o	/ cl	o i
8Ø	5Ø	P	P cl	P i	112	7Ø	p	Ø cl	p i
81	51	Q	Q cl	Q i	113	71	q	1 cl	q i
82	52	R	R cl	R i	114	72	r	2 cl	r i
83	53	S	S cl	S i	115	73	s	3 cl	s i
84	54	T	T cl	T i	116	74	t	4 cl	t i
85	55	U	U cl	U i	117	75	u	5 cl	u i
86	56	V	V cl	V i	118	76	v	6 cl	v i
87	57	W	W cl	W i	119	77	w	7 cl	w i
88	58	X	X cl	X i	12Ø	78	x	8 cl	x i
89	59	Y	Y cl	Y i	121	79	y	9 cl	y i
9Ø	5A	Z	Z cl	Z i	122	7A	z	: cl	z i
91	5B	°	° cl	° i	123	7B	é	; cl	é i
92	5C	ç	ç cl	ç i	124	7C	ù	< cl	ù i
93	5D	§	§ cl	§ i	125	7D	è	= cl	è i
94	5E	^	^ cl	^ i	126	7E	¨	> cl	¨ i
95	5F	_	_ cl	_ i	127	7F	Del	? cl	▩ i

cl pour affichage clignotant.
i pour affichage inversé.
/ Prim pour caractères primaires.
/ Alt pour caractères alternatifs.

Le changement du jeu Primaire au jeu Alternatif est obtenu par POKE -16369,Ø.

Le changement du jeu Alternatif au jeu Primaire par POKE -1637Ø,Ø.

* Les icônes activés par PRINT CHR$(27) remplacent cette colonne. Pour revenir au mode majuscules, faire PRINT CHR$(24).

LES CARACTERES ALPHANUMERIQUES ET LEURS CODES ASCII ET ECRAN

Dec	Hex	Caractère				Dec	Hex	Caractère		
		Clavier		Ecran				Clavier	Ecran	
128	8Ø	c à	NUL	à	N	16Ø	AØ	Espace	Espace	N
129	81	c A	SOH	A	N	161	A1	!	!	N
13Ø	82	c B	STX	B	N	162	A2	"	"	N
131	83	c C	ETX	C	N	163	A3	£	£	N
132	84	c D	EOT	D	N	164	A4	$	$	N
133	85	c E	ENQ	E	N	165	A5	%	%	N
134	86	c F	ACK	F	N	166	A6	&	&	N
135	87	c G	BEL	G	N	167	A7	'	'	N
136	88	c H	BS ←	H	N	168	A8	(	(	N
137	89	c I	HT →|	I	N	169	A9	)	)	N
138	8A	c J	LF ↓	J	N	17Ø	AA	*	*	N
139	8B	c K	VT ↑	K	N	171	AB	+	+	N
14Ø	8C	c L	FF	L	N	172	AC	,	,	N
141	8D	c M	CR	M	N	173	AD	-	-	N
142	8E	c N	SO	N	N	174	AE	.	.	N
143	8F	c O	SI	O	N	175	AF	/	/	N
144	9Ø	c P	DLE	P	N	176	BØ	Ø	Ø	N
145	91	c Q	DC1	Q	N	177	B1	1	1	N
146	92	c R	DC2	R	N	178	B2	2	2	N
147	93	c S	DC3	S	N	179	B3	3	3	N
148	94	c T	DC4	T	N	18Ø	B4	4	4	N
149	95	c U	NAK →	U	N	181	B5	5	5	N
15Ø	96	c V	SYN	V	N	182	B6	6	6	N
151	97	c W	ETB	W	N	183	B7	7	7	N
152	98	c X	CAN	X	N	184	B8	8	8	N
153	99	c Y	EM	Y	N	185	B9	9	9	N
154	9A	c Z	SUB	Z	N	186	BA	:	:	N
155	9B	Esc	ESC	°	N	187	BB	;	;	N
156	9C	c ç	FS	ç	N	188	BC	<	<	N
157	9D	c §	GS	§	N	189	BD	=	=	N
158	9E	c ^	RS	^	N	19Ø	BE	>	>	N
159	9F	c _	US	_	N	191	BF	?	?	?

c X pour Ctrl X.

N pour affichage NORMAL.

Dec	Hex	Caractère Clavier	Caractère Ecran		Dec	Hex	Caractère-écran Clavier	Caractère-écran //c		Caractère-écran II	
192	CØ	à	à	N	224	EØ	`	`	N	Esp	n
193	C1	A	A	N	225	E1	a	a	N	!	n
194	C2	B	B	N	226	E2	b	b	N	"	n
195	C3	C	C	N	227	E3	c	c	N	£	n
196	C4	D	D	N	228	E4	d	d	N	$	n
197	C5	E	E	N	229	E5	e	e	N	%	n
198	C6	F	F	N	23Ø	E6	f	f	N	&	n
199	C7	G	G	N	231	E7	g	g	N	'	n
2ØØ	C8	H	H	N	232	E8	h	h	N	(	n
2Ø1	C9	I	I	N	233	E9	i	i	N	)	n
2Ø2	CA	J	J	N	234	EA	j	j	N	*	n
2Ø3	CB	K	K	N	235	EB	k	k	N	+	n
2Ø4	CC	L	L	N	236	EC	l	l	N	,	n
2Ø5	CD	M	M	N	237	ED	m	m	N	-	n
2Ø6	CE	N	N	N	238	EE	n	n	N	.	n
2Ø7	CF	O	O	N	239	EF	o	o	N	/	n
2Ø8	DØ	P	P	N	24Ø	FØ	p	p	N	Ø	n
2Ø9	D1	Q	Q	N	241	F1	q	q	N	1	n
21Ø	D2	R	R	N	242	F2	r	r	N	2	n
211	D3	S	S	N	243	F3	s	s	N	3	n
212	D4	T	T	N	244	F4	t	t	N	4	n
213	D5	U	U	N	245	F5	u	u	N	5	n
214	D6	V	V	N	246	F6	v	v	N	6	n
215	D7	W	W	N	247	F7	w	w	N	7	n
216	D8	X	X	N	248	F8	x	x	N	8	n
217	D9	Y	Y	N	249	F9	y	y	N	9	n
218	DA	Z	Z	N	25Ø	FA	z	z	N	:	n
219	DB	°	°	N	251	FB	é	é	N	;	n
22Ø	DC	ç	ç	N	252	FC	ù	ù	N	<	n
221	DD	§	§	N	253	FD	è	è	N	=	n
222	DE	^	^	N	254	FE	¨	¨	N	>	n
223	DF	_	_	N	255	FF	DEL	▒	N	?	n

N pour NORMAL en Apple //e (minuscules) et Apple //c

n pour NORMAL en Apple II et PLUS (pas de minuscules sur l'écran).

Table d'équivalence des caractères français et américains

Code décimal	35	64	91	92	93	96	123	124	125	126
Code hexadécimal	23	4Ø	5B	5C	5D	6Ø	7B	7C	7D	7E
Caractères français	£	à	°	ç	§	`	é	ù	è	¨
Caractères américains	#	@	[	\	]	`	{	|	}	~

CONVERSION HEXADECIMALE/DECIMALE

de nombres de 4 chiffres hexadécimaux $H3 H2 H1 HØ

COEFF	H3	H2	H1	H0
0	0	0	0	0
1	4096	256	16	1
2	8192	512	32	2
3	12288	768	48	3
4	16384	1024	64	4
5	20480	1280	80	5
6	24576	1536	96	6
7	28672	1792	112	7
8	32768	2048	128	8
9	36864	2304	144	9
A	40960	2560	160	10
B	45056	2816	176	11
C	49152	3072	192	12
D	53248	3328	208	13
E	57344	3584	224	14
F	61440	3840	240	15

```
CONVERSION HEXADECIMALE/DECIMALE
LE NOMBRE DECIMAL EST OBTENU
EN FAISANT LA SOMME DES VALEURS
PRISES A L'INTERSECTION
DE LA LIGNE DU CHIFFRE HEXA
ET DE LA COLONNE DE LA POSITION
DE CE CHIFFRE DANS LE NOMBRE HEXA

    EXEMPLE
  $AFF6 DEVIENT 40960 (A EN H3)
              + 3840 (F EN H2)
              +  240 (F EN H1)
              +    6 (6 EN H0)
              ------
          SOIT: 45046
```

APPLESOFT

Les messages d'erreur dans un programme en APPLESOFT sont de la forme :

? message ERROR IN numéro de ligne

'message' est le nom de l'erreur.
Le numéro de ligne est celui de l'instruction dans laquelle une erreur a été rencontrée. (Les erreurs ne sont détectées dans un programme qu'au moment de l'exécution des instructions).

Dès l'erreur détectée, l'interpréteur Basic Applesoft provoque l'arrêt du programme et l'affichage du message. Les variables et les instructions ne sont pas modifiées mais le programme ne peut pas se poursuivre. Les compteurs de boucles FOR-NEXT sont mis à zéro et la trace des GOSUB est annulée.

Grâce à l'instruction ONERR GOTO et à un sous-programme de traitement des erreurs (attendues!) un programme peut malgré tout se poursuivre normalement.

Les instructions données en mode immédiat (sans numéro de ligne) peuvent aussi déclencher un message d'erreur, il ne comportera pas d'indication de numéro de ligne.

Chaque type d'erreur est associé à un code qui figure à l'adresse décimale 222 (ou $DE) au moment de l'erreur.

Le numéro de ligne où l'erreur s'est produite figure aux adresses 218 et 219 (ou $DA, $DB). La valeur du pointeur TXPTR dans l'instruction erronée figure aux adresses 220 et 221 (ou $DC, $DD). La valeur du pointeur du stack au moment de l'erreur est conservée à l'adresse 223 (ou $DF). Cette information doit être restaurée avant tout traitement d'erreur grâce au sous-programme suivant :

```
300   68      PLA
301   A8      TAY
302   68      PLA
303   A6 DF   LDX $DF
305   9A      TXS
306   48      PHA
307   98      TYA
308   48      PHA
309   60      RTS
```

Le sous-programme est inscriptible en mémoire par les instructions suivantes dans le programme :

```
Ø POKE 216,0 : POKE 768,104 : POKE 769,168 : POKE 770,104 :
POKE 771,166 : POKE 772,223 : POKE 773,154 : POKE 774,72 :
POKE 775,152 : POKE 776,72 : POKE 777,96
```

A l'adresse décimale 216 figure une indication d'activation ($8Ø) ou de désactivation ($ØØ) de l'instruction ONERR GOTO.

Le schéma de programmation de la prise en compte des erreurs avant l'arrêt du programme est le suivant :

```
   Ø  Mise en mémoire du sous-programme ci-dessus
   1  ONERR GOTO 1ØØØ : activation du système
  1Ø  Déroulement normal du programme
 999  END
1000  CALL 768 : exécution du sous-programme
1010  IF PEEK(222)=5 THEN Prise en compte de l'erreur n° 5
1020  Détection d'autres codes d'erreurs
1030  RESUME pour revenir à l'instruction erronée si nécessaire
```

Le sous-programme qui contrôle le déroulement en cas d'erreur est à l'adresse $D412. Il provoque l'exécution du sous-programme HANDLERR à l'adresse $F2E9 si ONERR a été activé. HANDLERR met en place les mémoires $DA à $DF.

Si ONERR n'a pas été utilisée, alors le sous-programme $D412 arrête l'exécution et affiche le message d'erreur.

Dans l'interpréteur APPLESOFT, la table des messages d'erreur est implantée à partir de l'adresse $D260 et leurs codes correspondent à leur position dans cette table.

```
NEXT WITHOUT FORSYNTAXRETURN WITHOUT GO
SUBOUT OF DATAILLEGAL QUANTITYOVERFLOWOU
T OF MEMORYUNDEF'D STATEMENTBAD SUBSCRIP
TREDIM'D ARRAYDIVISION BY ZEROILLEGAL DI
RECTTYPE MISMATCHSTRING TOO LONGFORMULA
TOO COMPLEXCAN'T CONTINUEUNDEF'D FUNCTIO
N ERROR IN
BREAK
```

MESSAGES D'ERREUR

Liste des messages d'erreurs par ordre alphabétique

Code	Message	Origine	Commentaire
1Ø7	?BAD SUBSCRIPT (mauvais indice)	DIM	Tentative d'appeler un élément de tableau d'indice supérieur à la limite fournie dans DIM. Exemple : DIM A(15) avec A(20) ou encore avec un nombre d'indices différent de celui spécifié dans DIM. L'applesoft dimensionne automatiquement à 11 les variables indicées non déclarées.
21Ø	CAN'T CONTINUE (on ne peut pas continuer)	CONT	Impossibilité de reprendre l'exécution d'un programme par CONT. En cas d'erreur ou d'insertion ou de modification d'une instruction. Dans certains on peut repartir avec GOTO numéro de ligne.
133	?DIVISION BY ZERO (division par zéro)	/Ø	Peut venir d'une variable non initialisée à une valeur différente de zéro.
	?EXTRA IGNORED (donnée de trop ignorée)	INPUT	Si les données (séparées par des virgules) sont en nombre supérieur à celui des variables prévues pour les recevoir. Le programme se poursuit pourtant.
191	?FORMULA TOO COMPLEX (formule trop complexe)	IF "chaîne de caractère" THEN	Le test ne peut être demandé plus de 2 fois dans un programme.
149	?ILLEGAL DIRECT (illégal en mode direct)	Mode direct	Les instructions INPUT GET DEF FN et DATA ne sont pas autorisées en mode direct.
53	?ILLEGAL QUANTITY (valeur erronée)	Fonctions maths	Le paramètre donné à une fonction dépasse les limites permises - l'indice d'une variable est négatif - l'argument de LOG est négatif ou nul - l'argument de SQR est négatif. A PUISSANCE B si A est négatif et B n'est pas entier.

Code	*Message*	*Origine*	*Commentaire*
53	?ILLEGAL QUANTITY	MID$ LEFT$ RIGHT$	La longueur ou l'indice de positionnement ne sont pas compris entre 1 et 255.
		CHR$	Le code n'est pas compris entre Ø et 255 (bornes incluses).
		ASC	Le caractère est de longueur Ø (vide).
		CALL	L'adresse n'est pas comprise entre -65535 et +65535.
		POKE	L'adresse n'est pas comprise entre -65535 et 65535 ; la valeur n'est pas comprise entre Ø et 255.
		HIMEM:	L'adresse n'est pas comprise entre -65535 et 65535.
		HPLOT	X,Y<Ø ou X>278 et Y>191
		DRAW	X,Y<Ø ou X>278 et Y>191
		PLOT,VLIN, HLIN	X,Y<Ø ou X,Y>39
		PDL	X<Ø ou X>255
		HTAB	X<Ø ou X>255
		VTAB	X<Ø ou X>24
		SPC	X<0 ou X>255
		TAB(	X<Ø ou X>255
		ON..GOTO ON..GOSUB	L'index ne doit pas dépasser 255 ou être inférieur à Ø. Si la valeur de l'index est nulle ou plus élevée que le nombre de numéros de ligne spécifiés, l'exécution continue à l'instruction suivante.
Ø	NEXT WITHOUT FOR (NEXT sans FOR correspondant)	FOR,NEXT	.Des boucles FOR-NEXT ont été mal imbriquées exemple : FOR X = 1 TO... FOR Y = 1 TO... PRINT X,Y NEXT : NEXT Y (Réécrire NEXT Y : NEXT X) .Il manque un FOR pour un NEXT isolé.

Code	Message	Origine	Commentaire
42	?OUT OF DATA (Data épuisées)	READ	Essai d'exécution d'un READ alors que toutes les données du DATA ont déjà été lues. Prévoir de tester un caractère de fin de données ou une variable de comptage ou faire RESTORE pour relire les données au début de DATA.
		RECALL STORE	Ne pas utiliser des variables dont le nom commence par RECALL ou STORE.
77	?OUT OF MEMORY (Mémoire épuisée)	DIM	Ne pas dépasser le nombre maximum d'indices : 88.
		GOSUB	Ne peut gérer plus de 24 niveaux d'appels imbriqués.
		HIMEM:	Ne pas la fixer trop basse
		LOMEM:	Ne pas la fixer trop haute ou inférieure à la valeur actuelle. Le programme est trop grand ou les variables sont trop nombreuses.
66	?OVERFLOW ERROR (dépassement de capacité)	Nombre réel	Résultat supérieur à 1.7E38 (Un nombre réel est mémorisé avec 1 octet d'exposant et 4 octets de mantisse. Si le résultat est inférieur à 2.9E-39, il équivaut à 0 sans message d'erreur.
		STR$	Si le nombre à convertir en chaîne de caractères est trop grand.
		VAL	Si la valeur absolue du nombre cherché est supérieure à 1E38 ou si le nombre contient plus de 38 chiffres (dont les zéros les plus à droite).

Code	*Message*	*Origine*	*Commentaire*
120	?REDIM'D ARRAY (tableau redimensionné)	DIM	Un même tableau ne peut être dimensionné deux fois. (On a repassé deux fois sur l'instruction DIM).
	?REENTER (refaire l'entrée de données)	INPUT	On a fourni une quantité alphanumérique ; il faut reprendre en redonnant toutes les valeurs *numériques* attendues par l'instruction INPUT.
22	?RETURN WITHOUT GOSUB (retour sans GOSUB)	RETURN	Un sous-programme a été placé après la fin logique du programme où END a été oublié.
			Dans le traitement d'une erreur, reprendre dans un sous-programme sans exécuter GOSUB.
176	?STRING TOO LONG (chaîne de caractères trop longue)	X$	Ne pas créer une chaîne par concaténation dont la longueur dépasse 255, si l'argument est une chaîne de longueur totale supérieure à 255.
		LEN VAL	
		PRINT	A$+B$ a plus de 255 caractères (écrire PRINT AB)
16	?SYNTAX (erreur de syntaxe)		Instruction incompréhensible pour l'interpréteur APPLESOFT. - parenthèses non appariées - caractères illégaux - mauvaise ponctuation - faute d'orthographe dans un mot clé.
		ASC	Sur "CTRL @" ou CHR$(Ø)
		CONT INPUT	Si une entrée de donnée en INPUT est interrompue par 'CTRL C' et que CONT est essayé pour repartir.
16	?SYNTAX (erreur de syntaxe)	DATA	Une chaîne de caractères contenant ? n'est pas acceptée.

Code	Message	Origine	Commentaire
16	?SYNTAX	DEL	Doit être suivi de 2 numéros de ligne dans l'ordre croissant
		FOR..NEXT	Ne pas utiliser une variable de type entier (%) comme indice de boucle.
		HGR HGRZ TEXT	Ne pas utiliser ces mots-clés comme premières lettres d'un nom de variable (Ils seront exécutés avant affichage de l'erreur).
		IF..THEN	Il manque THEN avec le IF correspondant.
		LIST, Q	Affiche le programme complet puis le message SYNTAX.
		RECALL STORE SHLOAD	Ne pas utiliser ces mots-clés comme premières lettres d'un nom de variable
		RESUME	Elle est rencontrée avant qu'une erreur se soit produite. Peut être une erreur fatale.
		N° d'instruction incorrect	Si la lettre O est tapée au lieu du chiffre Ø ou la lettre I au lieu du chiffre 1.
163	?TYPE MISMATCH (désaccord entre numérique et alphanumérique)	LET MID$ LEFT$ RIGHT$	- Une variable chaîne ne peut recevoir une valeur numérique et vice-versa - erreur de type d'arguments.
224	?UNDEF'D FUNCTION (fonction non définie)		Référence à une fonction pour laquelle il n'existe pas d'instruction DEF FN préalable
9Ø	?UNDEF'D STATEMENT	GOTO GOSUB ON. GOTO RUN THEN	Renvoient à un numéro de ligne inexistant.

MESSAGES D'ERREUR CONCERNANT LES FICHIERS DOS 3.3

Dans le système d'exploitation des disquettes chargé en MEV (dans une configuration de 48K), la table des messages d'erreurs est implantée à partir de l'adresse ØA971. Le premier message est 'Return','Bell','Return'. Les suivants sont :

```
LANGUAGE NOT AVAILABLERANGE ERRORWRI
TE PROTECTEDEND OF DATAFILE NOT FOUNDVOL
UME MISMATCHI/O ERRORDISK FULLFILE LOCKE
DSYNTAX ERRORNO BUFFERS AVAILABLEFILE TY
PE MISMATCHPROGRAM TOO LARGENOT DIRECT C
OMMAND
```

Dans la zone suivante ØAA3F à ØAA4F, sont enregistrées les positions du début de chaque message dans la table des messages précédents :

Ø	3	25	25	36	51	62	76	91	1ØØ	1Ø9	12Ø	132	152	17Ø	187	Dec
0	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15	Code
				Ø24	Ø33											Hexa

Exemple : le message 'WRITE PROTECTED' est le 4ème de la liste, son code d'erreur est 4 et le texte est mémorisé de l'adresse ØA971+Ø24 à l'adresse ØA971+Ø32.

Cette analyse permet de traduire en français les messages d'erreur envoyés par le système d'exploitation des disquettes. Ne vous en privez pas !

Code	*Message*	*Origine*	*Cause*
9	DISK FULL (disquette pleine)	SAVE WRITE	Le répertoire arrive à sa limite où tous les secteurs sont utilisés.
5	END OF DATA (fin des données)	INPUT	Les données sont insuffisantes pour satisfaire l'instruction INPUT.
		APPEND READ	Après cette instruction seule l'instruction WRITE est autorisée.
		POSITION READ	La position atteinte ne correspond à aucune donnée enregistrée.
		EXEC F,Rr	Si r correspond au 2ème champ *après* la fin du fichier !!
		READ F,Rr	Si r correspond à un enregistrement non encore effectué (codé Ø).
10	FILE LOCKED (fichier ou programme verrouillé)	SAVE DELETE BSAVE WRITE	Un fichier comportant une astérisque sur le CATALOG ne peut être modifié en ré-écriture (sauf avec APPEND).
6	FILE NOT FOUND (fichier inconnu)	LOAD RUN BLOAD BRUN DELETE	Le fichier demandé n'existe pas sur la disquette. Vérifiez l'orthographe de son nom.
13	FILE TYPE MISMATCH (désaccord sur le type du fichier)	LOAD BLOAD RUN BRUN	Un fichier de type T ou B ne peut être appelé par LOAD ou RUN et un fichier de type I ou A ne peut être appelé par BLOAD ou BRUN.
13	FILE TYPE MISMATCH (désaccord sur le type de fichier)	OPEN READ POSITION WRITE APPEND EXEC CLOSE	Ces commandes ne sont opérationnelles que sur un fichier de *données* (de type T).

Code	*Message*	*Origine*	*Cause*
		CHAIN	Un programme en Basic Applesoft ne peut être 'chaîné' par cette seule commande qui ne concerne que les programmes en Basic Integer.
8	I/O ERROR (erreur d'entrée/sortie)	Toute commande	- lecteur sans disquette - n° de connecteur (sans contrôleur) - disquette abimée - disquette non initialisée - porte du lecteur ouverte
		VERIFY	S'il y a une erreur après vérification qu'un fichier est correctement ou mal enregistré.
1	LANGUAGE NOT AVAILABLE (interpréteur Basic absent du système)	LOAD FP INT	Un programme en Basic ne peut être exécuté sans que l'interpréteur soit présent.
		APPLE][PLUS	N'a pas d'interpréteur BASIC INTEGER en standard.
		CARTE LANGAGE	Si la carte-mère contient en MEM un interpréteur, la carte langage pourra être chargée avec l'autre.
12	NO BUFFERS AVAILABLE (trop de fichiers ouverts en MEV)	MAXFILES n	Le nombre maximum est 16 (le système en utilise 1 pour chaque commande). Le nombre par défaut est 3.
15	NOT DIRECT COMMAND (commande directe illégale)	OPEN READ WRITE APPEND POSITION	Ne pas les utiliser en mode direct. - *Ecrire un programme* contenant ces commandes dans un PRINT.
14	PROGRAM TOO LARGE (programme trop grand)	LOAD RUN	Le HIMEM est trop bas (le SED compare le nombre de secteurs du programme avec l'octet de poids fort de HIMEM).

Code	*Message*	*Origine*	*Cause*
2,3	RANGE ERROR (valeur erronée)		Mn-Max
		V	Volume Ø-254
		D	Lecteur Ø-2
		S	Connecteur 1-7
		L	Longueur 1-32767
		R	Numéro Ø-32767
		B	Numéro Ø-32767
		A	Adresse Ø-65535
		MAXFILES	Nombre de fichiers ouverts 1-16
11	SYNTAX ERROR (erreur de syntaxe dans une commande du SED	INT EXEC	Commande sans paramètres, instruction BASIC non valide
		Aa	a ni négatif
		Ll	l ni > 65535
		IN#S PR#s	s ne peut être supérieur à 7.
7	VOLUME MISMATCH (désaccord de numéros de volume)	V	Le volume de la disquette courante est différent de celui de la disquette demandée. Si la demande est faite avec VØ, la détection de volume ne sera pas faite.
4	WRITE PROTECTED (protection en écriture)		L'encoche recouverte, les fichiers ne seront accessibles qu'en lecture.
			La disquette est peut-être placée à l'envers. La disquette SYSTEM MASTER est toujours protégée.
		CHAIN	Un programme en Basic Applesoft ne peut être 'chaîné' par cette seule commande qui ne concerne que les programmes en Basic Integer.
8	I/O ERROR (erreur d'entrée/sortie)	Toute commande	- lecteur sans disquette - n° de connecteur (sans contrôleur) - disquette abimée - disquette non initialisée - porte du lecteur ouverte

MESSAGES D'ERREUR FICHIERS SED - DOS 3.3

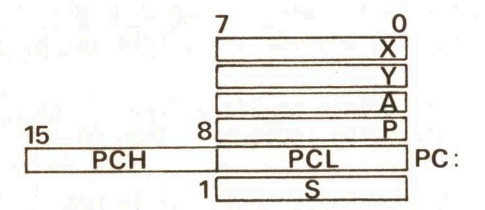
Code	*Message*	*Origine*	*Cause*
		VERIFY	S'il y a une erreur après vérification qu'un fichier est correctement ou mal enregistré.
1	LANGUAGE NOT AVAILABLE (interpréteur Basic absent du système)	LOAD FP INT	Un programme en Basic ne peut être exécuté sans que l'interpréteur soit présent.
12	NO BUFFERS AVAILABLE (trop de fichiers ouverts en MEV)	MAXFILES n	Le nombre maximum est 16 (le système en utilise 1 pour chaque commande). Le nombre par défaut est 3.
15	NOT DIRECT COMMAND (commande directe illégale)	OPEN READ WRITE APPEND POSITION	Ne pas les utiliser en mode direct. - *Ecrire un programme* contenant ces commandes dans un PRINT.
14	PROGRAM TOO LARGE (programme trop grand)	LOAD RUN	Le HIMEM est trop bas (le SED compare le nombre de secteurs du programme avec l'octet de poids fort de HIMEM).

Liste par code croissant.

Code	*Message*	*Cause*
2	**RANGE ERROR** (valeur erronée)	Les paramètres des commandes ont des limites :
	S	N° de connecteur 1-7
	D	N° de lecteur 1-2
	F	Nombre de champs Ø-65535
	R	N° d'enregistrement Ø-.
	B	N° d'octet Ø-.
	A	Adresse en MEV Ø-65535
	L	Longueur en octets Ø-65535
	E	Adresse en MEV Ø-65535
3	**NO DEVICE CONNECTED** (pas de périphérique)	Le périphérique appelé n'est pas branché.
4	**WRITE PROTECTED** (protection en écriture)	Si l'encoche de la disquette est recouverte, les fichiers sont protégés en écriture mais non en lecture.
5	**END OF DATA** (épuisement des données)	Le pointeur est hors du fichier. Voir position et READ.
6,7	**PATH NOT FOUND** (fichier introuvable)	Le nom d'accès est incomplet ou la disquette n'est pas la bonne.
8	**I/O ERROR** (erreur d'E/S)	Le lecteur est mal fermé ou ne contient pas de disquette ou la disquette est vierge ou formatée en DOS 3.2.
9	**DISK FULL** (disquette pleine)	La disquette est saturée de fichiers (51 par répertoire) ou ne dispose plus de blocs libres.
1Ø	**FILE LOCKED** (fichier verrouillé)	Le fichier est protégé donc on n'a pas le droit d'écrire dessus.
11	**INVALID OPTION** (option non valide)	L'option n'est pas disponible avec cette commande.
12	**NO BUFFERS AVAILABLE** (pas de zones-tampons disponibles)	La mémoire vive est saturée, elle ne peut ouvrir que huit fichiers ; de plus, les commandes CAT, APPEND, EXEC en utilisent aussi.
13	**FILE TYPE MISMATCH** (type de fichier incompatible)	La commande n'est pas faite pour ce type de fichier.

Code	*Message*	*Cause*
14	**PROGRAM TOO LARGE** (programme trop grand)	La MEV est saturée.
15	**NOT DIRECT COMMAND** (commande non directe)	Certaines commandes ne fonctionnent que dans un programme en Basic.
16	**SYNTAX ERROR**	Erreur de syntaxe dans la commande.
17	**DIRECTORY FULL**	Le répertoire est saturé.
18	**FILE NOT OPEN**	Le fichier n'a pas été ouvert.
19	**DUPLICATE FILENAME** (nom de fichier en double)	Les commandes CREATE et RENAME vérifient que le nouveau fichier n'existe pas déjà au répertoire.
2Ø	**FILE BUSY** (fichier occupé)	On ne peut pas effacer ou renommer ou cataloguer un fichier déjà ouvert.
21	**FILE(S) STILL OPEN** (fichiers encore ouverts)	Au cours d'une interruption par 'Ctrl C', des fichiers n'avaient pas été encore fermés.

REGISTRES INTERNES DU 65C02



registre d'index
registre d'index
accumulateur
registre d'état
compteur ordinal
pointeur de pile ('stack')

Détail du registre d'état P du microprocesseur

bit :	7	6	5	4	3	2	1	0
	N	V	/	B	D	I	Z	C

N signe
V débordement
5 inutilisé
B indicateur de BReaK
D mode décimal
I inhibition interruptions
Z résultat nul
C retenue

LANGAGE MACHINE

ADC : Addition avec retenue (Add with Carry) : A ← A + M + C ; on ajoute à l'accumulateur le contenu de la mémoire spécifiée plus le bit C de retenue ; opère en mode binaire ou décimal ; agit sur N, V, Z, C.

AND : ET logique : A ← A ∧ M ; fait l'opération bit à bit en accumulateur et mémoire ; agit sur N et Z.

ASL : Décalage à gauche (Arithmetic Shift Left) ; C ←[←]← 0 ; décale à gauche l'accumulateur ou une mémoire ; agit sur N, Z, C.

BCC : Branchement si pas de retenue (Branch on Carry Clear) ; si le bit C=0, on saute à l'instruction indiquée, sinon on continue en séquence.

BCS : Branchement si retenue (Branch on Carry Set) ; si le bit C=1, on saute à l'instruction indiquée ; sinon, on continue en séquence.

BEQ : Branchement si le résultat est nul (Branch on EQual) ; si le bit Z=1 (c'est-à-dire si le dernier résultat est 0 ou si la comparaison a donné l'égalité), on saute à l'instruction indiquée ; sinon, on continue en séquence.

BIT : Test de bits (BIt Test) $Z \leftarrow \Sigma\ \overline{Ai \wedge Mi}$, N ← M7, V ← M6 ; effectue le ET virtuel de l'accumulateur et de la mémoire spécifiée et positionne Z en conséquence ; en outre, les bits 7 et 6 sont copiés respectivement dans N et V sauf en mode immédiat.

BMI : Branchement si négatif (Branch on MInus) ; si le bit N=1, on saute à l'instruction indiquée, sinon, on continue en séquence.

BNE : Branchement si non égal à 0 (Branch on Not Equal) ; si le bit Z=0 (c'est-à-dire si le dernier résultat est différent de zéro ou si la dernière comparaison n'a pas donné l'égalité), on saute à l'instruction indiquée, sinon on continue en séquence.

BPL : Branchement si positif ou nul (Branch if PLus) ; si le bit N=0, on saute à l'instruction indiquée, sinon on continue en séquence.

BRA : Branchement inconditionnel (Branch Relative Always) ; on saute à l'adresse indiquée sans condition.

BRK : Interruption logicielle (BReak) ; met les bit B et I à 1 et simule une interruption (saut à l'adresse contenue en FFFF, FFFE).

BVC : Branchement si pas de débordement (Branch on Overflow Clear) ; si le bit V=0, on saute à l'instruction indiquée, sinon on continue en séquence.

BVS : Branchement si débordement (Branch on Overflow Set) ; si le bit V=1, on saute à l'instruction indiquée, sinon on continue en séquence.

CLC : Annuler la retenue (CLear Carry) ; force à Ø le bit C de retenue.

CLD : Annuler le mode décimal (CLear Decimal mode) ; force à Ø le bit D pour mettre l'unité arithmétique en fonctionnement binaire.

CLI : Autoriser les interruptions (CLear Interrupt Inhibit flag) ; force à Ø le bit d'inhibition des interruptions.

CLV : Annuler l'indicateur de débordement (CLear oVerflow flag) ; force à Ø le bit V.

CMP : Comparer avec l'accumulateur (ComPare with accumulator) ; A - M ; effectue la soustraction virtuelle : registre A - mémoire et positionne les indicateurs N, Z et C en conséquence.

CPX : Comparer avec X (ComPare with X) ; X - M ; effectue la soustraction virtuelle : registre X - mémoire et positionne les indicateurs N, Z et C en conséquence.

CPY : Comparer avec Y (ComPare with Y) ; Y - M ; effectue la soustraction virtuelle : registre Y - mémoire et positionne les indicateurs N, Z et C en conséquence.

DEA : Décrémenter l'accumulateur (DEcrement Accumulator) ; A ← A - 1 ; diminue de 1 le contenu de l'accumulateur ; agit sur N et Z.

DEC : Décrémenter en mémoire (DECrement Memory) ; M ← M - 1 ; diminue de 1 le contenu de la mémoire indiquée ; agit sur N et Z.

DEX : Décrémenter X (DEcrement X) ; X ← X - 1 ; diminue de 1 le contenu du registre X ; agit sur N et Z.

DEY : Décrémenter Y (DEcrement Y) ; Y ← Y - 1 ; diminue de 1 le contenu du registre Y et agit sur N et Z.

EOR : OU exclusif (Exclusive OR) ; effectue le OU exclusif entre l'accumulateur et la mémoire indiquée ; agit sur N et Z.

INA : Incrémenter l'accumulateur (Increment Accumulator) ; augmente de 1 le contenu de l'accumulateur et agit sur N et Z.

INC : Incrémenter en mémoire (INcrement Memory) ; augmente de 1 le contenu de la mémoire indiquée et agit sur N et Z.

INX : Incrémenter X (INcrement X) ; augmente de 1 le contenu de X et agit sur N et Z.

INY : Incrémenter Y (INcrement Y) ; augmente de 1 le contenu de Y et agit sur N et Z.

JMP : Saut inconditionnel (JuMP) ; PC adresse ; saute à l'adresse indiquée.

JSR : Appel d'un sous-programme (Jump to Sub-Routine) ; PC ↓ ; PC ← adresse ; sauve PC dans la pile (adresse de retour) puis saute à l'adresse indiquée.

LDA : Charger l'accumulateur (LOad Accumulator) ; A ← M ; met dans l'accumulateur le contenu de la mémoire spécifiée et agit sur N et Z.

LDX : Charge le registre X (LOad X register) ; X ← M ; met dans le registre X le contenu de la mémoire spécifiée et agit sur N et Z.

LDY : Charge le registre Y (LOad Y register) ; Y ← M ; met dans le registre Y le contenu de la mémoire spécifiée et agit sur N et Z.

LSR : Décalage à droite (Logical Shift Right) ; décale d'un bit vers la droite l'accumulateur ou une mémoire ; agit sur N, Z et C. Ø C.

NOP : Pas d'opération (No Operation) PC ← PC + 1 ; instruction muette s'exécutant en deux cycles.

ORA : OU inclusif (OR Accumulator) A ← A v M ; effectue le OU inclusif entre l'accumulateur et la mémoire indiquée ; agit sur N et Z.

PHA : Empiler A (Push A) A ↓ : (S) ← A ; S ← S - 1 ; met l'accumulateur en haut de la pile et met à jour le pointeur de pile.

PHP : Empiler P (PusH Processor status register) ; P ↓ : (S) ← P : S ← S - 1 ; met le registre P en haut de la pile et met à jour le pointeur de pile.

PHX : Empiler X (PusH X). X ↓ : (S) ← X : S ← S - 1 ; met le registre X au sommet de la pile et met à jour le pointeur de pile.

PHY : Empiler Y (PuH Y). Y ↓ : (S) ← Y : S ← S - 1 ; met le registre Y au sommet de la pile et met à jour le pointeur de pile ; affecte N et Z.

PLA : Dépiler A (PulL A) ; A ↑ : S ← S + 1 : A ← (S) ; met à jour le pointeur de pile et transfère vers A le contenu du haut de la pile ; affecte N et Z.

PLP : Dépiler P (PulL Processor status register) P ↑ : S ← S + 1 : P ← (S) ; met à jour le pointeur de pile et transfère le contenu du haut de la pile dans P ; affecte tous les indicateurs.

PLX : Dépiler vers X (PulL X register from stack) ; X ↑ : S ← S + 1 : X ← (S) ; met à jour le pointeur de pile et transfère vers le registre X le contenu du sommet de la pile.

PLY : Dépiler vers Y (PulL Y register from stack) ; Y ↑ : S ← S + 1 : Y (S) ; met à jour le sommet de la pile et transfère vers le registre Y le contenu du sommet de la pile.

ROL : Rotation à gauche (Rotate Left) ; décale d'un bit sur la gauche l'accumulateur ou une mémoire ; l'ancienne valeur du bit C rentre par la droite tandis que l'ancien bit 7 qui sort par la gauche vient remplacer C ; affecte N, Z et C.

ROR : Rotation à droite (ROtate Right) ; décale d'un bit à droite l'accumulateur ou une mémoire ; l'ancienne valeur du bit C rentre par la gauche tandis que l'ancien bit Ø qui sort par la droite vient remplacer C ; affecte N, Z et C.

RTI : Retour d'interruption (ReTurn from Interrupt) P ↑ ; PC ↑ ; retour du sous-programme d'interruption ; récupère sur la pile le PC et P qui y avaient été sauvés par le mécanisme d'interruption.

RTS : Retour de sous-programme (ReTurn from Sub-routine) PC ↑ ; récupère sur la pile l'ancien PC qui y avait été sauvé par le dernier JSR.

SBC : Soustraire avec retenue (Substract with Carry) ; A ← A - M - $\overline{C}$; on soustrait à l'accumulateur la mémoire spécifiée et aussi l'opposé du bit de retenue (l'emprunt) ; opère en mode binaire ou décimal et agit sur N, V, Z et C.

SEC : Mettre à 1 la retenue (SEt Carry flag) ; force à 1 le bit C.

SED : Mettre à 1 le mode décimal (SEt Decimal mode) ; force à 1 le bit D (influe donc sur ADC et SBC).

SEI : Masquer les interruptions (SEt Interrupt inhibit flag) ; force à 1 le bit I.

STA : Ranger l'accumulateur (STore Accumulator) M ← A ; transfère le contenu de l'accumulateur dans la mémoire spécifiée.

STX : Ranger le registre X (STore X register) M ← X ; transfère le contenu du registre X dans la mémoire indiquée.

STY : Ranger le registre Y (STore Y register) M ← Y ; transfère le contenu du registre Y dans la mémoire indiquée.

STZ : Met à zéro (Store Zero) M ← Ø ; donne un contenu nul à la mémoire indiquée.

TAX : Transférer A dans X ; X ← A ; agit sur N et Z.

TAY : Transférer A dans Y ; Y ← A ; agit sur N et Z.

TRB : Tester et mettre à zéro des bits de mémoire à l'aide de l'accumulateur (Test and Reset memory Bits with accumulator). $M \leftarrow \overline{A} \wedge M$; les bits 6 et 7 de la mémoire indiquée sont copiés en V et N, puis les bits à 1 dans l'accumulateur remettent à zéro les bits correspondant dans la mémoire M ; l'indicateur Z sera modifié en conséquence.

TSB : Tester et mettre à un des bits de mémoire à l'aide de l'accumulateur (Test and Set mémory Bits with accumulator) ; les bits 6 et 7 de la mémoire indiquée sont copiés en V et N, puis les bits à 1 dans l'accumulateur mettent à un les bits correspondants dans la mémoire M ; l'indicateur Z est modifié en conséquence.

TSX : Transférer S dans X ; $X \leftarrow S$; agit sur N et Z.

TXA : Transférer X dans A ; $A \leftarrow X$; agit sur N et Z.

TXS : Transférer X dans S ; $S \leftarrow X$; n'agit pas sur les indicateurs.

TYA : Transférer Y dans A ; $A \leftarrow Y$; agit sur N et Z.

Tableau de désassemblage

A partir d'un code à deux chiffres hexadécimaux H1HØ, ce tableau permet de trouver le code mnémonique et le mode d'adressage de l'instruction correspondante. *Exemple :* A9 → LDA Imm (ligne A, colonne 9). S'il n'y a pas de mode d'adressage, cela veut dire qu'il est inhérent ou relatif.

Le symbole ° signale une instruction propre au 65CØ2.

H1 \ HØ	Ø	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Ø	BRK	ORA ind,X			TSB° pgz	ORA pgz	ASL pgz		PHP	ORA imm	ASL A		TSB° abs	ORA abs	ASL abs	
1	BPL	ORA ind,Y	ORA° (pgz)		TRB° pgz	ORA pgz,X	ASL pgz,X		CLC	ORA abs,Y	INA° A		TRB° abs	ORA abs,X	ASL abs,X	
2	JSR abs	AND ind,X			BIT pgz	AND pgz	ROL pgz		PLP	AND imm	ROL A		BIT abs	AND abs	ROL abs	
3	BMI	AND ind,Y	AND° (pgz)		BIT° pgz,X	AND pgz,X	ROL pgz,X		SEC	AND abs,Y	DEA° A		BIT° abs,X	AND abs,X	ROL abs,X	
4	RTI	EOR ind,X				EOR pgz	LSR pgz		PHA	EOR imm	LSR A		JMP abs	EOR abs	LSR abs	
5	BVC	EOR ind,Y	EOR° (pgz)			EOR pgz,X	LSR pgz,X		CLI	EOR abs,Y	PHY°			EOR abs,X	LSR abs,X	
6	RTS	ADC ind,X			STZ° pgz	ADC pgz	ROR pgz		PLA	ADC imm	ROR A		JMP (abs)	ADC abs	ROR abs	
7	BVS	ADC ind,Y	ADC° (pgz)		STZ° pgz,X	ADC pgz,X	ROR pgz,X		SEI	ADC abs,Y	PLY°		JMP° abs (ind,X)	ADC abs,X	ROR abs,X	
8	BRA°	STA ind,X			STY pgz	STA pgz	STX pgz		DEY	BIT° imm	TXA		STY abs	STA abs	STX abs	
9	BCC	STA ind,Y	STA° (pgz)		STY pgz,X	STA pgz,X	STX pgz,Y		TYA	STA abs,Y	TXS		STZ° abs	STA abs,X	STZ° abs,X	
A	LDY imm	LDA ind,X	LDX imm		LDY pgz	LDA pgz	LDX pgz		TAY	LDA imm	TAX		LDY abs	LDA abs	LDX abs	
B	BCS	LDA ind,Y	LDA° (pgz)		LDY pgz,X	LDA pgz,X	LDX pgz,Y		CLV	LDA abs,Y	TSX		LDY abs,X	LDA abs,X	LDX abs,X	
C	CPY imm	CMP ind,X			CPY pgz	CMP pgz	DEC pgz		INY	CMP imm	DEX		CPY abs	CMP abs	DEC abs	
D	BNE	CMP ind,Y	CMP° (pgz)			CMP pgz,X	DEC pgz,X		CLD	CMP abs,Y	PHX°			CMP abs,X	DEC abs,X	
E	CPX imm	SBC ind,X			CPX pgz	SBC pgz	INC pgz		INX	SBC imm	NOP		CPX abs	SBC abs	INC abs	
F	BEQ	SBC ind,Y	SBC° (pgz)			SBC pgz,X	INC pgz,X		SED	SBC abs,Y	PLX°			SBC abs,X	INC abs,X	

LANGAGE MACHINE

Tableau d'assemblage des instructions suivant le mode d'adressage

Chaque case du tableau contient le code de l'instruction et le nombre de cycles nécessaires à l'exécution de l'instruction.

Mnémo	*Mode d'adressage*	*Imm*	*Abs*	*Pgz*	*Acc*	*Inh*	*(ind, X)*	*(ind) ,Y*
	Nombre d'octets	*2*	*3*	*2*	*1*	*1*	*2*	*2*
	Instructions							
A D C	A←A+M+C (1,3)	69 2	6D 4	65 3			61 6	71 5
A N D	A←A∧M (1)	29 2	2D 4	25 3			21 6	31 5
A S L	C ← [7 ← Ø] ← Ø		ØE 6	Ø6 5	ØA 2			
B C C	Brt si C=Ø (2)							
B C S	Brt si C=1 (2)							
B E Q	Brt si Z=1 (2)							
B I T	A∧M (4,5)	89 2	2C 4	24 3				
B M I	Brt si N=1 (2)							
B N E	Brt si Z=Ø (2)							
B P L	Brt si N=Ø (2)							
B R A	Brt Incond. (2)							
B R K	Arrêt					ØØ 7		
B V C	Brt si V=Ø (2)							
B V S	Brt si V=1 (2)							
C L C	Ø→ C					18 2		
C L D	Ø→ D					D8 2		
C L I	Ø→ 1					58 2		
C L V	Ø→ V					B8 2		
C M P	A - M	C9 2	CD 4	C5 3			C1 6	D1 5
C P X	X - M	EØ 2	EC 4	E4 3				
C P Y	Y - M	CØ 2	CC 4	C4 3				
D E A	A← A - 1				3A 2			
D E C	M← M - 1		CE 6	C6 5				
D E X	X← X - 1					CA 2		
D E Y	Y← Y - 1					88 2		
E O R	A← A ⊕ M	49 2	4D 4	45 3			41 6	51 5
I N A	A← A + 1				1A 2			
I N C	M← M + 1		EE 6	E6 5				
I N X	X← X + 1					E8 2		
I N Y	Y← Y + &					C8 2		
J M P	Saut incond.		4C 3					
J S R	Saut à sous-pg.		2Ø 6					
L D A	A← M (1)	A9 2	AD 4	A5 3			A1 6	B1 5
L D X	X← M (1)	A2 2	AE 4	A6 3				
L D Y	Y← M (1)	AØ 2	AC 4	A4 3				

Mnémo	Mode d'adressage / Nombre d'octets / Instructions	*Imm* 2	*Abs* 3	*Pgz* 2	*Acc* 1	*Inh* 1	*(ind, X)* 2	*(ind),Y* 2
L S R	Ø→[7 Ø]→[C] (1)		4E 6	46 5	4A 2			
N O P	PC ← PC + 1					EA 2		
O R A	A ← A v M	Ø9 2	ØD 4	Ø5 3			Ø1 6	11 5
P H A	Ms ← A,S ← S-1					48 3		
P H P	Ms ← P,S ← S-1					Ø8 3		
P H X	Ms ← X,S ← S-1					DA 3		
P H Y	Ms ← Y,S ← S-1					5A 3		
P L A	S ← S+1,A ← Ms					68 4		
P L P	S ← S+1,P ← Ms					28 A		
P L X	S ← S+1,X ← Ms					FA 4		
P L Y	S ← S+1,Y ← Ms					7A 4		
R O L	[7←Ø] ← [C] (1)		2E 6	26 5	2A 2			
R O R	[C] → [7→Ø] (1)		6E 6	66 5	6A 2			
R T I	Retour Interrup.					4Ø 6		
R T S	Retour sous-pg.					6Ø 6		
S B C	A ← A-M-$\overline{C}$ (1,3)	E9 2	ED 4	E5 3			E1 6	F1 5
S E C	C ← 1					38 2		
S E D	D ← 1					F8 2		
S E I	I ← 1					78 2		
S T A	M ← A		8D 4	85 3			81 6	91 6
S T X	M ← X		8E 4	86 3				
S T Y	M ← Y		8C 4	84 3				
S T Z	M ← ØØ		9C 4	64 3				
T A X	X ← A					AA 2		
T A Y	Y ← A					A8 2		
T R B	M ← $\overline{A}$ ∧ M (4)		1C 6	14 5				
T S B	M ← A v M (4)		ØC 6	Ø4 5				
T S X	X ← S					BA 2		
T X A	A ← X					8A 2		
T X S	S ← X					9A 2		
T Y A	A ← Y					98 2		

Notes (voir abréviations page 83)

(1) Ajouter 1 au nombre de cycles si changement de page.

(2) Ajouter 1 au nombre de cycles si le branchement aboutit dans la même page.
Ajouter 2 au nombre de cycles si le branchement aboutit dans une autre page.

(3) Ajouter 1 au nombre de cycles si en mode décimal.

(4) V est rendu égal au bit 6 de la mémoire avant l'exécution.
N est rendu égal au bit 7 de la mémoire avant l'exécution.

(5) Le mode d'adressage immédiat de l'instruction BIT ne change pas V et N.

JEU D'INSTRUCTIONS DU 65CØ2

Mnémo	Mode Adressage	Pgz,X	Pgz,Y	Abs,X	Abs,Y	Rel	(Abs)	Abs (i,x)	(Zpg)
	Nombre octets	2	2	3	3	2	2	3	2
	Reg P 7654321Ø NV BDIZC								
A D C	NV....ZC	75 4		7D 4	79 4				72 5
A N D	N.....Z.	35 4		3D 4	39 4				32 5
A S L	N.....ZC	16 6		1E 6					
B C C	....					9Ø 2			
B C S	....					BØ 2			
B E Q	....					FØ 2			
B I T	76....Z.	34 4		3C 4					
B M I	....					3Ø 2			
B N E	....					DØ 2			
B P L	....					1Ø 2			
B R A	....					8Ø 2			
B R K	...1.1..								
B V C	....					5Ø 2			
B V S	....					7Ø 2			
C L C	.......Ø								
C L D	....Ø...								
C L I	.....Ø..								
C L V	.Ø......								
C M P	N.....ZC	D5 4		DD 4	D9 4				D2 5
C P X	N.....ZC								
C P Y	N.....ZC								
D E A	N.....Z.								
D E C	N.....Z.	D6 6		DE 6					
D E X	N.....Z.								
D E Y	N.....Z.								
E O R	N.....Z.	55 4		5D 4	59 4				52 5
I N A	N.....Z.								
I N C	N.....Z.	F6 6		FE 6					
I N X	N.....Z.								
I N Y	N.....Z								
J M P	....						6C 3	7C 3	
J S R	....								
L D A	N.....Z.	B5 4		BD 4	B9 4				B2 5
L D X	N.....Z.		B6 4		BE 4				
L D Y	N.....Z.	B4 4		BC 4					
L S R	Ø.....ZC	56 6		5E 6					
N O P	....								

Mnémo	*Mode Adressage* / *Nombre octets* / *Reg P* 76543210 NV BDIZC	*Pgz,X*	*Pgz,Y*	*Abs,X*	*Abs,Y*	*Rel*	*(Abs)*	*Abs (i,x)*	*(Zpg)*
		2	2	3	3	2	2	3	2
O R A	N.....Z.	15 4		1D 4					12 5
P H A	....								
P H P	....								
P H X	....								
P H Y	....								
P L A	N.....Z.								
P L P	NV.1DIZC								
P L X	N.....Z.								
P L Y	N.....Z.								
R O L	N.....ZC	36 6		3E 6					
R O R	N.....ZC	76 6		7E 6					
R T I	NV.1DIZC								
R T S	....								
S B C	NV....ZC	F5 4		FD 4	F9 4				F2 5
S E C	1								
S E D	....1...								
S E I	.....1..								
S T A	....	95 4		9D 5	99 5				92 5
S T X	....		96 4						
S T Y	....	94 4							
S T Z	....	74 4		9E 5					
T A X	N.....Z.								
T A Y	N.....Z.								
T R B	......Z.								
T S B	......Z.								
T S X	N.....Z.								
T X A	N.....Z.								
T X S	....								
T Y A	N.....Z.								

Abréviations

X	Registre d'index X	+	Addition	6	Bit 6 de la mémoire
Y	Registre d'index Y	-	Soustraction	$\underline{7}$	Bit 7 de la mémoire
A	Accumulateur	Λ	ET	$\overline{A}$	Complément de A
M	Mémoire	V	OU		
Ms	Mémoire de la pile	⊕	OU Exclusif		

LES "COMMENT...?"

1 - Positionnement du curseur

Maintenir le curseur en face de la question jusqu'à ce que la réponse soit correcte.

La position verticale du curseur est **mémorisée** avant la saisie et restaurée pour une nouvelle saisie.

Si la réponse est incorrecte, elle est **effacée.**

Exemple :

```
5  HOME
6 Q$ = "REPONDEZ PAR O(UI OU N(ON "
10  PRINT Q$;: GOSUB 100
20  INPUT "";R$
30  IF R$ = "O" THEN  VTAB PV: PRINT Q$;"OUI"
40  IF R$ = "N" THEN  VTAB PV: PRINT Q$;"NON"
45  IF  LEFT$ (R$,1) <  > "O" AND  LEFT$ (R$,1) <
    > "N" THEN  VTAB PV: PRINT Q$;: CALL  - 868:
    GOTO 20
50  PRINT "FIN": PRINT : GOTO 10
100  REM
200 PV =  PEEK (37) + 1
210  RETURN
```

PV est la position verticale (ligne)

CALL-868 efface le reste de la ligne depuis l'endroit où se trouve le curseur.

2 - Simuler INPUT X$

La chaîne lue s'inscrit de $200 (512) à $2FF (768)

APPLESOFT	Assembleur
CALL-10964	JSR $D52C

Tous les caractères sont acceptés jusqu'à concurrence de 255 mais 'CTRL X' annule la ligne et 'Return' valide l'entrée.

3 - Empêcher le listing d'un programme

POKE 2049,Ø : POKE 2050,Ø

met à zéro le pointeur du début de la deuxième ligne d'instructions.

Pour retrouver sa valeur exacte, rechercher le premier octet ØØ de fin de la première ligne d'instructions et ajouter 1 à l'adresse de cet octet.

Exemple :

```
]LIST

10  REM  COMMENT N0 3
20  PRINT : END

]POKE2049,0:POKE2050,0

]LIST


]CALL -151
                   14Ø8
*800.81F

0800- 00 00 00 0A 00 B2 20 43
0808- 4F 4D 4D 45 4E 54 20 4E
0810- 30 20 33 00 1C 08 14 00
0818- BA 3A 80 00 00 00 0A 00
```

Les commandes NEW et FP annulent aussi ce pointeur sans effacer le programme.

4 - Mettre deux programmes bout-à-bout en DOS 3.3

a) Charger le programme de tête en mémoire vive.

b) Modifier le pointeur de début de programme pour qu'il pointe après l'octet ØØ de la dernière instruction du programme de tête.

c) Charger le programme de queue en mémoire vive.

d) Modifier le pointeur de début de programme pour qu'il pointe au début du programme de tête.

TEXTTAB pointeur de début de programme $67,$68
PRGEND pointeur de fin de programme $AF,$BØ

```
]NEW

]100 REM PG DE QUEUE

]SAVE PQ
]NEW

]10 REM PG DE TETE           a

]CALL -151

*AF.B0

00AF- 15
00B0- 08
*800.815

0800- 00 12 08 0A 00 B2 20 50
0808- 47 20 44 45 20 54 45 54
0810- 45 00 00 00 64 0A
*67:12 08                    b

*3D0G

]LOAD PQ                     c
]CALL -151

*67:01 08                    d

*3D0G

]LIST

10  REM  PG DE TETE
100  REM  PG DE QUEUE
```

5 - Empêcher l'accès au clavier

En dehors du blocage de la touche 'RESET' (cf 6) il faut prévoir de neutraliser aussi la touche 'CTRL C' qui provoque l'interruption du programme en cours avec le message:

BREAK IN n° de l'instruction où l'arrêt a été déclenché.

La solution proposée utilise **le traitement d'erreur** : 'CTRL C' correspond au code d'erreur n° 255 et une fois détectée, l'erreur est annulée par RESUME :

```
1 ON ERR GO TO 1000
1000 IF PEEK(222) = 255 THEN RESUME
```

6 - Toutes les commandes sont interprétées comme un RUN (DOS 3.3)

POKE 214,128

Une valeur supérieure ou égale à 128 dans l'adresse 214 ou $D6 a un effet irréversible sur toutes les commandes ou instructions Basic, elles sont transformées en RUN. Sont épargnées les commandes d'accès aux programmes sur disquettes.

Faire PR#6 pour réinitialiser le système.

- En PRODOS, l'effet de POKE 214,128 est tout autre !

7 - Inhiber la touche 'RESET'

L'effet de 'Ctrl Reset' sur le système dépend du contenu des mémoires $3F2 et $3F3.

L'adresse contenue dans ces mémoires est celle vers laquelle se branchera le système si 'Reset' est tapé.

Adresses		*Valeurs* par défaut		*PRODOS*	
Dec	*Hex*	*Dec*	*Hex*		
1010 1011	3F2 3F3	191 157	BF 9D	00 BE	retour à BASIC (arrêt du programme en cours)
1012	3F4	56	38	1B	OR exclusif de (1011) et #A5

La valeur de l'octet d'adresse $3F4 est obtenue par CALL-1169 ($FB6F) puis PRINT PEEK (1012). Si cette valeur n'est pas égale au OR exclusif de (1011) et £A5, alors il y aura un redémarrage à froid.

a) Inhibition : (le programme en cours ne s'arrêtera pas avec 'Reset').

```
* 3F2 : ØØ Ø3 A6        DOS 3.3

* 3ØØ : 2Ø EA Ø3        JSR $Ø3EA (DOS)
* 3Ø3 : 2Ø 98 D8        JSR $D898 (CONT)
* 3Ø5 : 4C D2 D7        JSR $D7D2 (NEWSTT)
```

b) Inhibition de tout le système (après avoir tapé 'Reset')

```
* 3F2 : ØØ Ø3 A6
* 3ØØ : 4C ØØ Ø3
```

Le redémarrage de tout le système n'est possible qu'après coupure du courant.

c) La touche 'Reset' fait redémarrer le système comme si on mettait sous tension :

il suffit d'un POKE 1012,Ø

		PRODOS
d) Désinhibition	* 3F2 : BF 9D 38	* 3F2 : ØØ BE 1B

e) Retour au moniteur 3F2 : 69 FF 5A

8 - Attendre un caractère au clavier

a) 1Ø X = PEEK(-16384) : IF X < 128 THEN 1Ø
   2Ø POKE - 16368,Ø : X$ = CHR$(X-128)

b) 1Ø WAIT -16384,128 : X = PEEK(-16384)-128 : POKE - 16368,Ø

c) 10 GET X$

d) 1Ø CALL-756 (RDKEY)

9 - Modifier l'affichage d'un listing de programme en Basic

(en mode 4Ø colonnes exclusivement)

- POKE 33,33

La fenêtre d'écran est réduite à 33 colonnes de largeur. La commande LIST affiche les instructions **sans marge.**

- TEXT annule la commande précédente

- Le caractère : permet d'introduire une indentation de ligne d'instruction.

- POKE 33,28
  facilite le cadrage des REM : la disposition à l'enregistrement n'est pas modifiée par LIST.

- TEXT ou POKE 33,4Ø pour revenir au mode standard.

10 - GOTO calculé par l'intermédiaire de l'ampersand &

On écrit & expression.

Le sous-programme est mémorisé à partir de $3ØØ. Donc les adresses $3F5 $3F6 et $3F7 doivent être pré-enregistrées avec l'instruction JMP $3ØØ pour que l'interpréteur se branche sur l'adresse $3ØØ dès qu'il rencontrera &.

* 3F5 : 4C ØØ Ø3

ou bien :

10 POKE 1013,76 : POKE 1014,Ø : POKE 1015,3

Sous-programme d'évaluation de l'expression et de branchement à la ligne calculée

```
*300L
0300-   20 7B DD    JSR   $DD7B
0303-   20 52 E7    JSR   $E752
0306-   20 1A D6    JSR   $D61A
0309-   90 03       BCC   $030E
030B-   4C 41 D9    JMP   $D941
030E-   A2 5A       LDX   #$5A
0310-   4C 12 D4    JMP   $D412
```

$DD7B	FRMEVL	évaluation de l'expression le résultat va dans FAC
$E752	GETADR	conversion FAC en valeur entière le résultat va dans $50,$51
$D61A	FNDLIN	recherche si la ligne calculée fait partie du programme
$D941	GOTO+	saut à la ligne trouvée
$D412	ERROR	erreur éventuelle avec code #5A = 9Ø UNDEF'D STATEMENT

11 - Imprimer avec D décimales

a) DEF FNF(X) = INT(X*1ØΛD)/10ΛD

au lieu d'imprimer X, on imprimera FNF(X)

Note : L'instruction PRINT d'un nombre réel n'affiche pas les zéros les plus à droite de la partie fractionnaire ni les zéros les plus à gauche de la partie entière.

Si Ø.Ø1 ⩽ |X| < 999 999 999.2 le nombre est en notation virgule fixe, sinon il est sous forme mantisse, exposant sx.xxx xxx xxEstt

s est le signe
. est la séparation entre partie entière et partie fractionnaire
E veut dire 10 à la puissance
x et T sont des chiffres de Ø à 9

b) Arrondir à D décimales

La fonction INT(X) a pour résultat le plus petit entier inférieur à X, ce qui pose quelque problème si X est négatif.

```
Ainsi  ?INT(-5.3)
       -6
```

Il faut donc tenir compte du signe de X en arrondissant.

```
Ainsi   ?INT(ABS(-5.3))/SGN(-5.3)
        -5
DEF FN AR(X) = INT(ABS(X)*10^D+.5)/10^D*SGN(X)
  Faire PRINT FNAR(X)
```

c) Tronquer à D décimales avec la notation flottante

```
10 X$ = STR$(X)
2Ø FOR I = 1 TO LEN(X$):IF MID$(X$,I,1)<>"E"THEN NEXT I
3Ø FOR J = 1 TO I-1:IF MID$(X$,J,1)<>"."THEN NEXT J
4Ø IF J+D< = I-1 THEN N=J+D:GO TO 6Ø
5Ø N = I-1
6Ø PRINT LEFT$(X$,N)+MID$(X$,I)
```

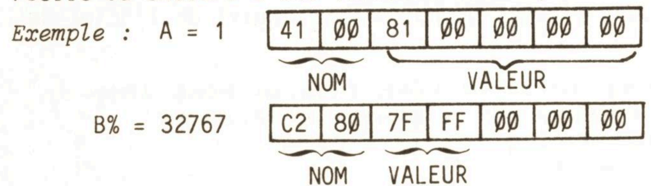
12 - Justifier à droite dans une zone de C caractères

```
A$ = STR$(FN AR(X))
C$ = REM C caractères 'espace'
PRINT RIGHT$(C$+A$,C)
```

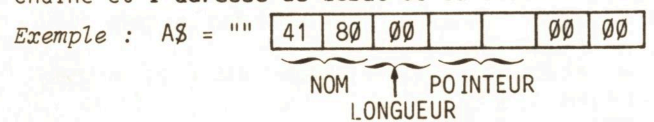
13 - Connaître l'adresse d'une variable

Il faut distinguer les variables numériques simples des variables alphanumériques.

Après les deux octets représentant les deux premières lettres du **nom**, le système réserve 5 octets pour conserver la **valeur** réelle ou entière d'une variable numérique.



Le cas des variables alphanumériques dont la valeur est une chaîne de caractères est différent puisque dans les 5 octets suivant le nom de la variable, on trouve **la longueur** de la chaîne et **l'adresse de début** de la chaîne.



L'adresse cherchée est celle de la valeur d'une variable **numérique** donc du pointeur de valeur.

L'adresse d'une chaîne de caractères est celle contenue dans les octets +1 et +2 par rapport au pointeur de valeur.

Ce pointeur de valeur (VARPNT) est mémorisé dans $83, $84 (131,132) et contient celui de la dernière variable traitée par l'APPLESOFT.

En APPLESOFT on écrit :

```
10 X=A : REM ON recherche l'adresse de A
20 A=PEEK(131) + 256*PEEK(132)
30 PRINT A
40 A=X : REM on rétablit la valeur de A
```

En langage machine, on peut se servir de la routine PTRGET d'adresse $DFE3 pour récupérer dans l'accumulateur et le registre Y, les poids faibles et forts du pointeur de la variable dont on cherche l'adresse.

Grâce à l'opérateur & suivi du nom de la variable, on entre dans ce sous-programme en langage machine qui renvoie aux adresses 778($30A) et 779($30B) la valeur cherchée.

```
*300L

0300-    20 E3 DF     JSR    $DFE3
0303-    8D 0A 03     STA    $030A
0306-    8C 0B 03     STY    $030B
0309-    60           RTS
```

Dans un tableau de valeurs numériques entières, la valeur n'occupe que deux octets pour chaque variable indicée. Le pointeur de valeur sera utilisé directement : l'adresse de la variable indicée contient l'octet de poids fort, suivi de l'octet de poids faible.

```
            1  POKE 1013,76: POKE 1014,0: POKE 1015,3
            2 X = 0:L = 0:P = 0:A$ = "APPLESOFT"
            5  DIM A%(100):A%(1) = 32767
            10  & A%(1)
            30 X =  PEEK (778) +  PEEK (779) * 256
            40  PRINT X
            45  PRINT 256 *  PEEK (X) +  PEEK (X + 1)
            50 A$ =  LEFT$ (A$,5)
            60  & A$
            70 X =  PEEK (778) +  PEEK (779) * 256
Longueur    80 L =  PEEK (X)
Pointeur    90 P =  PEEK (X + 1) +  PEEK (X + 2) * 256
            100  PRINT L,P;" ";
            110  FOR X = P TO P + L - 1
Chaîne      120  PRINT  CHR$ ( PEEK (X));: NEXT X
```

```
         ]RUN
Adresse  2365
Valeur   32767
         5                38395 APPLE
```

14 - Listing sur imprimante

PR£1
LIST

Avec l'imprimante IMAGE WRITER il faut demander un affichage simultané sur l'écran par
PRINT CHR$(9);"I"

Si les lignes d'instructions dépassent 30 caractères, il faut modifier le nombre de caractères par ligne éditée sur l'imprimante pour éviter le format classique (image de l'écran).

Après l'impression, on désactive l'imprimante par
PRINT CHR$(9);"R"

L'ordre PR£Ø désactive aussi.

15 - Changer de page d'écran

POKE -16299,0	affiche la page n° 2	*CØ55
POKE -163ØØ,Ø	affiche la page n° 1	*CØ54
POKE -163Ø4,Ø	affiche en graphique	*CØ5Ø
POKE -163Ø3,Ø	affiche en texte	*CØ51
POKE -16297,Ø	affiche en HGR sans effacement	*CØ57
POKE -16298,Ø	affiche en GR sans effacement	*CØ56
POKE -163Ø2,Ø	graphique sur tout l'écran	*CØ53
POKE -163Ø1,Ø	graphique et 4 lignes de texte	*CØ52

16 - Modifier la fenêtre d'écran-texte

TEXT règle la fenêtre aux valeurs maxima

Largeur	: $21(33)	: WNDWDTH	= $28(4Ø)
Marge gauche	: $2Ø(32)	: WNDLFT	= $Ø(Ø)
Marge haute	: $22(34)	: WNDTOP	= $$Ø(Ø)
Marge basse	: $23(35)	: WNDBTM	= $18(24)

POKE 33, largeur	comprise entre 1 et 4Ø
POKE 32, marge gauche	marge gauche + largeur inférieure à 39
POKE 34, marge haute	comprise entre Ø et 23
POKE 35, marge basse	supérieure à la marge haute et inférieure à 24

La marge gauche ne se positionne qu'après un 'Return' (PRINT)

17 - Faire afficher les icones (caractères Souris)

```
10   REM ESSAI DES CAR. SOURIS
20   REM   Déclenchement des S/P
21   REM  de gestion des 80COL:
22   PRINT  CHR$ (4);"PR£3"
25   REM   Affichage élargi
26   REM  (en 40 colonnes):
27   PRINT  CHR$ (17): REM CTRL/Q
40   REM  Mode Standard
41   PRINT  CHR$ (24)
42 D = 64: GOSUB 100
50   REM  Mode SOURIS
51   PRINT  CHR$ (27)
52 D = 64: GOSUB 100
60   REM  Mode standard
61   PRINT  CHR$ (24)
62 D = 80: GOSUB 100
70   REM  Mode SOURIS
71   PRINT  CHR$ (27)
72 D = 80: GOSUB 100
80   PRINT : NORMAL : END
100  FOR C = D TO 15 + D
115  INVERSE
120  PRINT  CHR$ (C);
121  NORMAL : PRINT " ";
125  IF C = 70 THEN  PRINT  CHR$
     (8);: REM  coureur!
130  NEXT C
140  PRINT
200  RETURN
```

18 - Utiliser la SOURIS

```
10   HOME
20   REM   Mettre la SOURIS
21   REM    en-ligne :
30   PRINT  CHR$ (4);"PR£4"
31   REM   en mode transparent :
32   PRINT  CHR$ (1)
38   REM    Rétablir la sortie
39   REM  en vidéo :
40   PRINT  CHR$ (4);"PR£0
45   REM   Saisir les données
47   REM   de la souris :
50   PRINT  CHR$ (4);"IN£4"
60   INPUT "";X,Y,S
65   REM   Afficher la position
67   REM et l'état de la souris:
```

```
70  VTAB 10: PRINT X;"    ",Y;"
    ",S"    "
74  REM  Etats du bouton
75  REM S=1 maintenu enfoncé
77  REM S=2 vient d'etre préssé
79  REM S=3 vient d'etre
80  REM        relaché
81  REM S=4 encore relaché
83  REM S < 0 si une touche du
84  REM clavier a été pressée
85  IF S > 0 THEN 60
86  REM  Saisir les données
87  REM   depuis le clavier :
90  PRINT  CHR$ (4);"IN£0"
95  REM  Mettre la souris
98  REM   hors-service :
100  PRINT  CHR$ (4);"PR£4": PRINT
      CHR$ (0)
105  REM Rétablir la sortie
106  REM  vidéo normale :
110  PRINT  CHR$ (4);"PR£0"
120  END
```

Pour éviter que l'instruction n°60 INPUT"";X,Y,S n'efface la ligne courante, il faut la faire précéder de 62 VTAB 23:HTAB 39.

19 - Transférer une image en mémoire auxiliaire

```
*301L
0301-   A9 00        LDA   £$00
0303-   85 3C        STA   $3C
0305-   A9 20        LDA   £$20
0307-   85 3D        STA   $3D
0309-   A9 F8        LDA   £$F8
030B-   85 3E        STA   $3E
030D-   A9 3F        LDA   £$3F
030F-   85 3F        STA   $3F
0311-   A9 00        LDA   £$00
0313-   85 42        STA   $42
0315-   AD 00 03     LDA   $0300
0318-   85 43        STA   $43
031A-   38           SEC
031B-   20 11 C3     JSR   $C311
031E-   60           RTS
031F-   00           BRK
0320-   00           BRK
0321-   00           BRK
0322-   00           BRK
0323-   00           BRK
```

Ce sous-programme, à appeler par CALL 769, prédispose les registres A1H,L et A2H,L pour transférer l'image graphique située entre $2000 et $3FFF. Le registre A4H,L est chargé avec le contenu de $300 ; cette adresse est le paramètre du transfert : à quelle page en mémoire auxiliaire est destinée l'image. Par exemple, POKE 768,32 : CALL 769 transfère en $2000 de la MEV auxiliaire, la page HGR.

Ce sous-programme se sert de MOVEAUX, sous-programme en MEM, situé en $C311, en positionnant la retenue C à 1 avant l'appel.

20 - Protéger un INPUT avec une valeur par défaut

La valeur par défaut est d'une longueur 1 caractère.

```
10  REM  SAISIE PAR DEFAUT
20 DE$ = "O": REM   VALEUR PAR DEFAUT
30  PRINT "QUESTION? ";DE$;
40 PH =  PEEK (36):PV =  PEEK (37) + 1
41  IF PV > 23 THEN PV = 23: REM  ATTENTION AU SCROLL

42  CALL  - 1008: REM  RECUL D'UNE POSITION
50  INPUT "";RE$
60  IF RE$ = "" THEN RE$ = DE$
70  HTAB PH: VTAB PV: PRINT RE$
```

Lorsque l'INPUT s'exécute, on voit le curseur clignoter sur la valeur par défaut. Si en réponse on tape 'Return', la valeur prise sera celle par défaut. Si l'on veut entrer une autre valeur, on tape cette valeur sur celle présentée, puis 'Return'.

21 - Prévoir la taille d'un programme

En gros un programme occupe autant d'octets qu'il renferme de caractères, puisqu'il est stocké tel quel, comme chaîne de caractères sauf les MOTS-CLES qui sont remplacés par un code en 1 octet.

En ce qui concerne les variables, chaque variable numérique simple réelle ou entière occupe 7 octets, chaque chaîne occupe (7 + longueur) octets.

Un tableau occupe $x(n+1) + 2d + 3$
n est la taille du tableau (y compris l'élément n° Ø)
x = 5 (nombres réels)
x = 2 (entiers)
x = 3 (chaînes de caractères)
d nombre de dimensions

On gagne de la place en mémoire en supprimant tous les blancs inutiles, en mettant plusieurs instructions par ligne, en évi-

tant les REM, en utilisant des variables plutôt que des constantes.

Utilisez les GOSUB dès qu'il faut faire appel plusieurs fois à une séquence d'instructions identiques.

22 - Faire jouer de la musique à l'APPLE

Un air est défini par une liste de paires I,J

I est la hauteur de la note ou sa fréquence

J est la durée de cette note (à quel tempo, ronde, blanche, noire, croche ? etc.).

Un programme, écrit en langage machine, permet de stimuler le haut parleur par LDA $C030, à intervalles réguliers :

- le registre X est initialisé avec la valeur I, avant un 'bip', et diminue jusqu'à 0, jusqu'au prochain 'bip' ;
  Plus I est faible, plus la fréquence, donc la hauteur, est élevée.
- le registre Y diminue aussi et son passage par zéro fait diminuer J, la durée, laquelle, en atteignant zéro, va provoquer la fin de l'exécution d'une note.

GAMME : (avec le programme ci-dessous). Valeurs de I :
255, 242, 230, 216, 204, 192, 182, 172, 162, 152, 144, 136, 128,
128, 121, 115, 108, 102, 96, 91, 86, 81, 76, 72, 68, 64,
64, 60, 57, 54, 51, 48, 45, 43, 40, 38, 36, 34, 32
SOL, SOL# , LA, LA# , SI, DO, DO# , RE, RE# , MI, FA, FA# , SOL

```
0302-    AD 30 C0    LDA   $C030
0305-    88          DEY
0306-    D0 05       BNE   $030D
0308-    CE 01 03    DEC   $0301
030B-    F0 09       BEQ   $0316
030D-    CA          DEX
030E-    D0 F5       BNE   $0305
0310-    AE 00 03    LDX   $0300
0313-    4C 02 03    JMP   $0302
0316-    60          RTS
```

Pour jouer un air, il faut appeler ce sous-programme pour chacune des notes successivement.

En Basic, les paires I,J sont lues sur un fichier DATA jusqu'aux valeurs 0,0.

Le programme ci-dessus est mémorisé en début de programme, par des instructions POKE A,V.

Exemple : deux petites musiques "synthétisées"

```
10  REM  MUSIQUE
20  POKE 770,173: POKE 771,48: POKE
     772,192: POKE 773,136: POKE
     774,208: POKE 775,5: POKE 77
     6,206: POKE 777,1: POKE 778,
     3: POKE 779,240: POKE 780,9:
      POKE 781,202
30  POKE 782,208: POKE 783,245: POKE
     784,174: POKE 785,0: POKE 78
     6,3: POKE 787,76: POKE 788,2
     : POKE 789,3: POKE 790,96: POKE
     791,0: POKE 792,0
40  READ I,J: IF J = 0 THEN 70
50  POKE 768,I: POKE 769,J: CALL
     770
60  GOTO 40
70  IF F = 1 THEN  END
80 F = 1: INPUT " ENCORE UNE?";R$
90  GOTO 40
100  DATA  114,120,144,60,114,255
     ,1,120,128,120,144,60,128,12
     0,114,60,144,120,171,255,228
     ,255,0,0
200  DATA  0,160,128,255,152,40,1
     71,80,192,40,228,255,1,40,0,
     160,192,255,192,40,171,80,15
     2,40,128,255,0,0
```

23 - Faire dessiner, tourner, agrandir une forme

Le codage d'une forme peut être simplifié en utilisant un octet pour chaque vecteur élémentaire tracé :
- dirigé vers le haut → 4
- dirigé vers la droite → 5
- dirigé vers le bas → 6
- dirigé vers la gauche → 7

Les vecteurs successifs sont rangés dans une zone choisie par l'utilisateur : on l'appelle une table de forme.

Elle doit contenir dans
- le 1er octet : le nombre de forme (1 par exemple)
- le 3ème et le 4ème : l'offset pour repérer le début de la 1ère forme (Ø4 ØØ pour une forme)

puis les vecteurs de la forme.

Il faut préciser en début de programme, l'adresse de début de cette table de forme. Cette adresse est rangée en $E8 ou 232 et

$E9 ou 233 avec la partie basse de l'adresse en $E8 (poids faibles).

En Basic, la succession des vecteurs du dessin rangée en DATA avec Ø comme fin de liste, est lue et mémorisée à partir du 5ème octet de la table (si une seule forme est prévue).

La forme choisie dans l'exemple suivant est un pétale "stylisée".

Une fleur avec un pétale comme forme définie

```
]LIST

10  HGR
20  HCOLOR= 3
25  REM  LA TABLE DES FORMES EST
     A L'ADRESSE $300 OU 768
30  POKE 232,0: POKE 233,3
35  REM   UNE SEULE FORME ... UN
      PETALE
40  POKE 768,1: POKE 769,0: POKE
     770,4: POKE 771,0
42  RESTORE :T = 0
43  READ D: POKE 772 + T,D: IF D =
     0 THEN 48
45 T = T + 1: GOTO 43
48 X = 140:Y = 80
50  SCALE= 3
52  REM  VOICI UNE FLEUR
55  FOR R = 0 TO 64 STEP 4
58  ROT= R
60  DRAW 1 AT X,Y
70  NEXT R
80  END
100  DATA  4,4,4,5,4,4,4,5,4,4,4,
     5,4,4,4,5,4,5,4,5,4,5,4,5,5,
     5,4,5,5,5,4,5,5,5,4,5,5,5
200  DATA  6,6,6,7,6,6,6,7,6,6,6,
     7,6,6,6,7,6,7,6,7,6,7,6,7,7,
     7,6,7,7,7,6,7,7,7,6,7,7,7 ,0

300  REM  LA DIRECTION D'UN VECTE
     UR ELEMENTAIRE VAUT 4,5,6,7
     POUR HAUT,DROITE,BAS,GAUCHE
     RESPECTIVEMENT
```



INDEX DES "COMMENT...?"

(Les numéros sont les numéros des "Comment...?")

PAGES ZERO A HUIT

Adresses MEV utilisées par tous les sous-programmes de base en MEM :

- le programme Monitor ;
- la gestion de l'affichage en 8Ø colonnes ;
- les sous-programmes d'E/S : série, souris, lecteur.

L'interpréteur Applesoft en MEM utilise certaines adresses de la page **Zéro** que nous détaillons plus loin.

Les **astérisques** * signalent des adresses utilisées par le système d'exploitation des disquettes.

Page ZERO

Dec	*Hex*	*Nom*	*Rôle*
Ø,1	ØØ,Ø1	LOCØ,LOC1	vecteur utilisé pour l'autodémarrage de la disquette.
2-5	ØØ,Ø5		- interpréteur Applesoft.
6-9	Ø6-Ø9	---------	non-utilisées.
1Ø-24	ØA-18		- interpréteur Applesoft.
25-31	19-1F	---------	non utilisées.
32,33 34,35	2Ø,21 22,23	WNDLFT,WDTH WNDTOP,BTM	marge gauche, largeur, marge haute et basse de la fenêtre.
36,37	24,25	CH,CV	coordonnées du curseur en 4Ø col.
38,39	26,27*	GBASL,GBASH	contiennent l'adresse de base d'une ligne graphique obtenue avec GBASCALC d'après l'acc.
4Ø,41	28,29*	BASL,BASH	contiennent l'adresse de base d'une ligne de texte calculée par BASCALC d'après l'acc.
42,43	2A,2B*	BAS2L,BAS2H	adresse de base d'une ligne utilisée en cas de défilement de l'image.

Dex	*Hex*	*Nom*	*Rôle*
44,45	2C,2D *	H2,V2	paramètres de tracé de verticales et horizontales graphiques.
44,45	2C,2D *	LMNEM,RMNEM	codes de mnémoniques (trois caractères sur deux octets) pour le désassemblage.
46	2E *	MASK	ØF (lignes paires), FØ (impaires) comme masque de couleurs GR.
46,47	2E,2F *	FORMAT,LENGTH	servent au désassemblage des opérandes.
48	3Ø	COLOR	indicateur de couleur pour deux lignes adjacentes.
49	31	MODE	indicateur d'opération des commandes au Monitor (. + - Ø).
5Ø	32	INVFLG	masque d'affichage Normal/ Inverse (ou Flash).
51	33	PROMPT	caractère de sollicitation affiché par GETLN avant une saisie.
52	34	YSAV	index Y dans l'analyse des commandes au Monitor.
53	35 *	YSAV1	index Y sauvegardé avant COUNTZ.
54,55	36,37 *	CSWL,CSWH	contiennent l'adresse de la routine de sortie de caractères.
56,57	38,39 *	KSWL,KSWH	contiennent l'adresse de la routine d'entrée de caractères. *Exemple :* clavier FD1B (sans le SED).
58,59	3A,3B *	PCL,PCH	pour la sauvegarde du compteur ordinal avant BREAK par exemple.
6Ø,61	3C,3D *	A1L,A1H	Registres temporaires pour les paramètres des sous-progr. MOVE, MOVEAUX, VERIFY.
62,63	3E,3F *	A2L,A2H	
64,65	4Ø,41 *	A3L,A3H	
66,67	42,43 *	A4L,A4H	
68,69	44,45 *	A5L,A5H	

Dec	*Hex*	*Nom*	*Rôle*
68	44 *	MACSTAT	Etat de la machine après BRK.
69	45 *	ACC	Accumulateur après un BRK.
7Ø	46 *	XREG	Registre X après un BRK.
71	47 *	YREG	Registre Y après un BRK.
72	48 *	STATUS	Registre P après un BRK.
73	49	SPNT	Registre S après un BRK.
74-77	4A-4D *		Système d'exploitation.
78,79	4E,4F	RNDL,RNDH	Compteur incrémenté pendant KEYIN. Sert d'amorce à RND.
8Ø-255	5Ø-FF *		- interpréteur Applesoft.

Page UN

Dec	*Hex*	*Nom*	*Rôle*
256-511	1ØØ-1FF	Pile	pile dont le haut est pointé par le registre S.

Page DEUX

Dec	*Hex*	*Nom*	*Rôle*
512-758	2ØØ-2F8	IN	tampon contenant les caractères entrés avec GETLN. La fin d'une ligne est matérialisée par RC.

Page TROIS

Dec	*Hex*	*Nom*	*Rôle*
768-975	3ØØ-3CF	----------	disponibles.
976-1ØØ7	3DØ-3EF *		- système d'exploitation.
1ØØ5,1ØØ6	3ED,3EE		registres utilisés par XFER.
1ØØ8,1ØØ9	3FØ,3F1	BRKV	vecteur de reprise après BRK (normalement $59,$FA).

Dec	*Hex*	*Nom*	*Rôle*
1010,1011	3F2,3F3	SOFTEV	vecteur de redémarrage à chaud après 'Ctrl Reset'.
1012	3F4	PWREDUP	égal à (3F3 EOR £A5) pour ne pas provoquer de redémarrage à froid après 'Ctrl Reset'.
1013-1015	3F5-3F7	AMPERV	instruction de sortie du Basic par &.
1016-1018	3F8-3FA	USRADR	instruction de sortie du Monitor par 'Ctrl Y'.
1019-1021	3FB-3FD	NMI	n'est pas utilisé en APPLE II C.
1022,1023	3FE,3FF	IRQLOC	vecteur de prise en charge d'une interruption masquable.

Pages QUATRE, CINQ, SIX, SEPT

Dec	*Hex*	*Nom*	*Rôle*
1024	400	LINE 1	début de la mémoire d'écran-texte ou écran-GR (basse résolution) 24 lignes de 40 caractères ou 48 lignes de 40 dominos de couleur.
1024-1063	400-427	écran	ligne 0 de texte ou 0,1 de GR.
1064-1103	428-44F	écran	ligne 8 de texte ou 16,17 de GR.
1104-1143	450-477	écran	ligne 16 de texte ou 32,33 de GR.
1144	478	ROMSTATE	indicateur temporaire de l'état de la MEM.
1145-1151	478+s		mémoires MEV accessibles aux interfaces connectés en "s".
1147	47B	OLDCH	précédente valeur de CH utilisée par l'affichage en 80 colonnes (C300).
1152-1191	480-4A7	écran	ligne 1 de texte ou 2,3 de GR.
1192-1231	4A8-4CF	écran	ligne 9 de texte ou 18,19 de GR.

Dec	*Hex*	*Nom*	*Rôle*
1232-1271	4DØ-4F7	écran	ligne 17 de texte ou 34,35 de GR.
1272	4F8	TEMP1	registre temporaire pour CTLCHAR.
1273-1279	4F8+s		mémoires MEV accessibles aux interfaces connectés en "s".
1275	4FB	VMODE	octet décrivant le mode opératoire (Basic ou Pascal, Icones).
128Ø-1319	5ØØ-527	écran	Ligne 2 de texte ou 4,5 de GR.
132Ø-1359	528-54F	écran	ligne 1Ø de texte ou 2Ø,21 de GR.
136Ø-1399	55Ø-577	écran	ligne 18 de texte ou 36,37 de GR.
14ØØ	578	TEMPA	registre temporaire pour SCROLL.
14Ø1-14Ø7	578+s		mémoires MEV accessibles aux interfaces connectées en "s".
14Ø3	57B	OURCH	position horizontale du curseur en affichage 8Ø colonnes.
14Ø8-1447	58Ø-5A7	écran	ligne 3 de texte ou 6,7 en GR.
1448-1487	5A8-5CF	écran	ligne 11 de texte ou 12,13 GR.
1488-1527	5DØ-5F7	écran	ligne 19 de texte ou 2Ø,21 GR.
1528	5F8	TEMPY	registre temporaire pour SCROLL.
1529-1535	5F8+s		mémoires MEV accessibles aux interfaces connectées en "s".
1531	5FB	OURCV	position verticale du curseur en affichage 8Ø colonnes.
1536-1575	6ØØ-627	écran	ligne 4 de texte ou 8,9 GR.
1576-1615	628-64F	écran	ligne 12 de texte ou 24,25 GR.
1616-1655	65Ø-677	écran	ligne 2Ø de texte ou 4Ø,41 GR.
1656-1663	678+s		mémoires MEV accessibles aux interfaces connectées en "s".
1659	67B	VFACTV	indicateur (bit 7) d'état d'activité de l'affichage en 8Ø colonnes.

Dec	*Hex*	*Nom*	*Rôle*
1664-1703	680-6A7	écran	ligne 5 de texte ou 10,11 GR.
1704-1743	6A8-6CF	écran	ligne 13 de texte ou 26,27 GR.
1744-1783	6D0-6F7	écran	ligne 21 de texte ou 42,43 GR.
1784-1791	6F8+s		mémoires MEV accessibles aux interfaces connectées en "s".
1787	6FB	XCOORD	coordonnée X utilisé par GOTOXY.
1792-1831	700-727	écran	ligne 6 de texte ou 12,13 GR.
1832-1871	728-74F	écran	ligne 14 de texte ou 28,29 GR.
1872-1911	750-777	écran	ligne 22 de texte ou 44,45 GR.
1912	778	DEVN0	$n0 : n° de l'interface active courante x 16.
1913-1919	778+s		mémoires MEV accessibles aux interfaces connectées en "s".
1915	77B	NXTCUR	le prochain curseur à afficher.
1920-1959	780-7A7	écran	ligne 7 de texte ou 14,15 GR.
1960-1999	7A8-7CF	écran	ligne 15 de texte ou 30,31 GR.
2000-2039	7D0-7F7	écran	ligne 23 de texte ou 46,47 GR.
2040	7F8	MSLOT	$Cn : n° de l'interface utilisant l'espace d'adresses C8-- (c'est C3 gérant les 80 col.).
2041-2047	7F8+s		mémoires MEV accessibles aux interfaces connectées en "s".

Rappel sur les interfaces intégrées dans l'APPLE II C

Connecteur "s"	*Type d'interface*	*Périphérique*	*Adresses MEV utilisées*
1	Série	Imprimante série	91, 479, 4F9, 579, 5F9, 6F9, 779, 7F9, 478-47D (MEVAUX)
2	Série	Modem	47A, 4FA, 57A, 5FA, 6FA, 77A, 7FA, 47C-47F (MEVAUX)

Connecteur "s"	*Type d'interface*	*Périphérique*	*Adresses MEV utilisées*
3	Video-8Øcol	Ecran	47B, 57B, 5FB, 67B, 6FB, 77B, 7FB.
4	Souris	Souris/manettes	478, 4F8, 578, 5F8, 47C, 4FC, 57C, 5FC, 67C, 67C, 77C.
5	-	-	47D, 4FD, 57D, 5FD, 67D, 6FD, 77D, 7FD.
6	Contrôleur disquette	Lecteur intégré	Ø3, 26, 27, 2B, 3C, 3D, 4Ø, 41, 4F, 3ØØ- 356, 7DB.
7	-	Lecteur externe	

Page HUIT - mémoire vive auxiliaire

Dec	*Hex*	*Nom*	*Rôle*
2Ø48-	8ØØ-	THBUF	tampon d'entrée pour les interfaces série (Modem).

LES ADRESSES DE COMMUTATION DES MEMOIRES

Pour déterminer l'état de la machine, il existe douze indicateurs binaires ; chacun d'eux est associé à un mode opératoire déterminé concernant la sélection donc la commutation des mémoires :

- **MEV** mémoire vive ou **RAM** ;
- **MEM** mémoire morte ou **ROM** ;
- **P** principale, **X** auxiliaire.

Mode	*Ø*	*1*	*Indicateur*	*Adresse*
Lecture	MEV P	MEV X	**RDRAMRD**	CØ13
Ecriture	MEV P	MEV X	**RDRAMWRT**	CØ14
P.Zéro, Pile	MEV P	MEV X	**RDALTZP**	CØ16
Banc DØ-DF	1	2	**RDLCBNK2**	CØ11
Mémoire DØ-FF	MEM	MEV	**RDLCRAM**	CØ12
Mémoire 8Øc	NON	OUI	**RD8ØSTORE**	CØ18
Gestion 8Øc	NON	OUI	**RD8ØVID**	CØ1F
Caractères	primaire	alternatif	**ALTCHARSET**	CØ1E
Ecran	graphique	texte	**RDTEXT**	CØ1A
Mode écran	plein	mixte	**RDMIX**	CØ1B
Page écran	1	2	**RDPAGE2**	CØ1C
Page 8Øcol	MEV P	MEV X	**RDPAGE2**	CØ1C
Résolution	basse	haute	**RDHIRES**	CØ1D
Accès E/S	OUI	NON	**RDIOUDIS**	CØ7E
DoubleHR	NON	OUI	**RDDHIRES**	CØ7F

Le contenu de chaque indicateur est obtenu par une instruction de lecture à son adresse. C'est le bit 7 qui donne l'état de sortie du "commutateur logiciel" associé. Chaque commutateur dispose de deux entrées, une entrée de mise à 1 et une entrée de mise à Ø. Ces entrées sont stimulées grâce à des instructions d'écriture dans des adresses spécifiques de l'espace CØØØ-CØFF.

L'astérisque * marque les cas où il faut deux instructions d'écriture successives pour obtenir le résultat.

Commutateur logiciel	*Nom* *Mise à Ø*	*Adresse*	*Nom* *Mise à 1*	*Adresse*
Lecture	CLRRAMRD	CØØ2	SETRAMRD	CØØ3
Ecriture	CLRRAMWRT	CØØ4	SETRAMWRT	CØØ5
P.Zéro,pile	SETSTDZP	CØØ8	SETALTZP	CØØ9
Banc DØ-DF	LCBANK1	CØ8B*	LCBANK2	CØ83* MEV lect/écr.
M. DØ-FF	ROMIN	CØ82	lec/écrMEV	CØ83* banc 2
	lectMEM	CØ8A	lec/écrMEV	CØ8B* banc 1
	écrMEVb2	CØ81*	lecMEVb2	CØ8Ø banc 2
	écrMEVb1	CØ89*	lecMEVb1	CØ88 banc 1

Commutateur logiciel	*Nom* *Mise à Ø*	*Adresse*	*Nom* *Mise à 1*	*Adresse*
Mémoire8Øc.	CLR8ØSTORE	CØØØ	SET8ØSTORE	CØØ1
Gestion8Øc.	CLR8ØVID	CØØC	SET8ØVID	CØØD
Caractères	CLRALTCHAR	CØØE	SETALTCHAR	CØØF
Ecran	TXTCLR	CØ5Ø	TXTSET	CØ51
Mode	MIXCLR	CØ52	MIXSET	CØ53
Page Ecran	TXTPAGE1	CØ54	TXTPAGE2	CØ55
Page 8Øcol	PAGE1P	CØ54	PAGE1X	CØ55 (RD8ØSTORE à 1)
Résolution	LORES	CØ56	HIRES	CØ57
Accès E/S	ENIOU	CØ7F	IOUDIS	CØ7E
DoubleHR	CLRDHIRES	CØ5F	SEDHIRES	CØ5E (RDIOUDIS à 1)

ADRESSES ENTRE CØØØ et CFFF
(zone d'E/S et des commutateurs)

Adresses *Dec*	*Hex*	*Nom*	*Rôle*	***L** = lire, **E** = écrire*
-16384	CØØØ	**KBD**	Données provenant du clavier (b7=1 dès qu'une touche est tapée).	L
-16384	CØØØ	**CLR8ØSTORE**	Mise à Ø de 8ØSTORE ; cet état rend à PAGE2 son rôle de commutateur des pages 1 et 2 (texte ou GR).	E
-16383	CØØ1	**SET8ØSTORE**	Mise à 1 de 8ØSTORE ; PAGE2 servira de commutateur entre les pages d'affichage principale et auxiliaire.	E
-16382	CØØ2	**CLRRAMRD**	Mise à Ø de RAMRD pour lire en mémoire principale.	E
-16381	CØØ3	**SETRAMRD**	Mise à 1 de RAMRD pour lire en mémoire auxiliaire.	E
-1638Ø	CØØ4	**CLRRAMWRT**	Mise à Ø de RAMWRT pour écrire en mémoire principale.	E
-16379	CØØ5	**SETRAMWRT**	Mise à 1 de RAMWRT pour écrire en mémoire auxiliaire.	E
-16378	CØØ6		Réservée.	
-16377	CØØ7		Réservée	
-16376	CØØ8	**SETSTDZP**	Mise à Ø de ALTZP pour utiliser la pile et la page Zéro de la mémoire principale.	E
-16375	CØØ9	**SETALTZP**	Mise à 1 de ALTZP pour utiliser la pile et la page Zéro de la mémoire auxiliaire.	E
-16374	CØØA		Réservée.	
-16373	CØØB		Réservée.	
-16372	CØØC	**CLR8ØVID**	Mise à Ø de 8ØVID pour désactiver les sous-programmes d'affichage 8Ø colonnes.	E
-16371	CØØD	**SET8ØVID**	Mise à 1 de 8ØVID pour activer l'affichage en 8Ø colonnes.	E
-1637Ø	CØØE	**CLRALTCHAR**	Mise à Ø de ALTCHAR pour afficher le jeu primaire (caractères clignotants).	E
-16369	CØØF	**SETALTCHAR**	Mise à 1 de ALTCHAR pour avoir le jeu alternatif (tous : N ou I).	E
-16368	CØ1Ø	**KBDSTRB**	Echantillonnage du clavier pour saisir une nouvelle touche (mise à Ø du bit 7 de KBD).	E
-16367	CØ11	**RDLCBNK2**	Indicateur pour voir quel est le banc utilisé en DØ-DF.	L
-16366	CØ12	**RDLCRAM**	Indicateur pour voir quel type de mémoire est lu entre DØ et FF.	L
-16365	CØ13	**RDRAMRD**	Indicateur pour voir si la mémoire auxiliaire est lisible.	L

Adresses Dec	Hex	*Nom*	*Rôle*	*L = lire, E = écrire*
-16364	CØ14	**RDRAMWRT**	Indicateur pour savoir si l'écriture en mémoire auxiliaire est autorisée.	
-16363	CØ15	**RSTXINT** ou **MOUXINT**	Demande d'interruption par le mvt horizontal de la souris.	L
-16362	CØ16	**RDALTZP**	Indicateur de la mémoire contenant la pile et la page Zéro courantes.	L
-16361	CØ17	**RSTYINT** ou **MOUYINT**	Demande d'interruption par le mvt vertical de la souris.	L
-1636Ø	CØ18	**RD8ØSTORE**	Indicateur de l'état de 8ØSTORE.	L
-16359	CØ19	**VBLINT**	Indicateur associé à l'interruption due à VBL (effacement vertical).	
-16358	CØ1A	**RDTEXT**	Indicateur de l'utilisation de l'écran en texte ou graphique.	L
-16357	CØ1B	**RDMIX**	Indicateur de mode mixte ou non.	L
-16356	CØ1C	**RDPAGE2**	Indicateur pour savoir quelle page est affichée.	L
-16355	CØ1D	**RDHIRES**	Indicateur de la résolution graphique.	L
-16354	CØ1E	**RDALTCHAR**	Indicateur du jeu de caractères.	L
-16353	CØ1F	**RD8ØVID**	Indicateur d'activation des sous-programmes de gestion de l'affichage sur 8Ø colonnes.	L
-16352 à -16337	CØ2Ø à CØ2F		Réservées.	
-16336	CØ3Ø	**SPKR**	Changement d'état du signal binaire envoyé sur le haut-parleur.	L
-16335 à -16321	CØ31 à CØ3F		Réservées.	
-1632Ø	CØ4Ø	**RDXYMSK**	Indicateur de masquage de XØ et YØ.	L
-16319	CØ41	**RDVBLMSK**	Indicateur de masquage de VBL.	L
-16318	CØ42	**RDXØEDGE**	Interruption due au front descendant de XØ.	L
-16317	CØ43	**RDYØEDGE**	Interruption due au front descende YØ.	L
-16316 à -16313	CØ44 à CØ47		Réservées.	L
-16312	CØ48	**RSTXY** ou **MOUCLR**	Remet à zéro les indicateurs d'interruption par XØ/YØ.	L
-16311 à -163Ø5	CØ49 à CØ4F		Réservées.	
-163Ø4	CØ5Ø	**TXTCLR**	Mise à Ø de TEXT pour faire afficher des graphiques.	L/E

ADRESSES ENTRE CØØØ et CFFF
(zone d'E/S et des commutateurs)

Adresses Dec	Hex	*Nom*	*Rôle*	*L = lire, E = écrire*
-163Ø3	CØ51	**TXTSET**	Mise à 1 de TEXT pour faire afficher du texte.	L/E
-163Ø2	CØ52	**MIXCLR**	Mise à Ø du mode mixte (plein g.).	L/E
-163Ø1	CØ53	**MIXSET**	Mise à 1 du mode mixte pour visualiser quatre lignes de texte sous l'image graphique.	L/E
-163ØØ	CØ54	**TXTPAGE1**	Mise à Ø de PAGE2 pour afficher la page 1 ($4ØØ ou $2ØØØ).	L/E
-16299	CØ55	**TXTPAGE2**	Mise à 1 de PAGE2 pour sélectionner la page 2 si 8ØSTORE est à Ø ou la page 1 de la MEV X si 8ØSTORE à 1.	
-16298	CØ56	**LORES**	Mise à Ø de HIRES pour dessiner en basse résolution (4Øx48).	L/E
-16297	CØ57	**HIRES**	Mise à 1 de HIRES pour dessiner en haute résolution (28Øx192).	L/E
-16296	CØ58		Réservé si l'accès E/S est empêché par IOUDIS à 1.	
-16296	CØ58	**DISXY** ou **MOUDSBL**	Masquage des interruptions dues à XØ/YØ.	L/E
-16295	CØ59		Réservé si IOUDIS à 1.	
-16295	CØ59	**ENBXY** ou **MOUENBL**	Autorise les interruptions dues à XØ/YØ.	L/E
-16294	CØ5A		Réservé à IOUDIS à 1.	
-16294	CØ5A	**DISVBL**	Masquage des interruptions du VBL.	
-16293	CØ5B		Réservé si IOUDIS à 1.	
-16293	CØ5B	**ENVBL**	Autorise les interruptions de VBL.	L/E
-16292	CØ5C		Réservé si IOUDIS à 1.	
-16292	CØ5C	**XØEDGE**	Interruption sur le front montant de XØ.	L/E
-16291	CØ5D		Réservé si IOUDIS à 1.	
-16291	CØ5D	**XØEDGE**	Interruption sur le front descendant de XØ.	L/E
-1629Ø	CØ5E	**SETDHIRES**	La double haute résolution est activée si IOUDIS est à 1.	L/E
-1629Ø	CØ5E	**YØEDGE**	Interruption sur le front montant de YØ.	L/E
-16289	CØ5F	**CLRDHIRES**	Mise à Ø du mode double haute résolution si IOUDIS est à 1.	L/E
-16289	CØ5F	**YØEDGE**	Interruption sur le front descendant de YØ.	
-16288	CØ6x		Réservées si en écriture.	E
-16288	CØ6Ø	**RD8ØSW**	Indicateur de l'interrupteur 8Ø/4Ø. Est à 1 s'il est en position basse (4Ø).	L

Adresses Dec	Hex	*Nom*	*Rôle*	*L = lire, E = écrire*
-16287	CØ61	**RDBTNØ**	Indicateur de l'état du bouton-poussoir n° Ø et de la touche POMME OUVERTE (blanche).	L
-16286	CØ62	**RDBTN1**	Indicateur de l'état du bouton-poussoir n° 1 et de la touche POMME FERMEE (noire).	L
-16285	CØ63	**RD63** ou **MOUBUT**	Indique si le bouton de la souris est relâché.	L
-16284	CØ64	**PDLØ**	Etat de sortie du monostable relié à la manette de jeux n° Ø.	L
-16283	CØ65	**PDL1**	Etat de sortie du monostable relié à la manette de jeux n° 1.	L
-16282	CØ66	**MOUX1**	Indique si le signal de direction X1 de la souris est à 1.	L
-16281	CØ67	**MOUY1**	Indique si le signal de direction Y1 de la souris est à Ø.	L
-1628Ø à -16273	CØ68 à CØ6F		Réservées en écriture et en lecture.	
-16272	CØ7Ø	**PTRIG** et **VBLCLR**	Remise à Ø du monostable pour débuter la lecture des manettes, remise à Ø de VBLINT.	E/L
-16271 à -16259	CØ71 à CØ7D		Réservées.	
-16258	CØ7E	**RDIOUDIS**	Même effet que PTRIG si IOUDIS à 1. Indicateur (b7) de l'état de IOUDIS.	L
-16258	CØ7E	**SETIOUDIS** ou **IOUDIS**	Mise à 1 de IOUDIS pour inhiber les accès E/S des adresses CØ58 à CØ5F et permettre la double haute résolution.	E
-16257	CØ7F	**RDDHIRES**	Indicateur du mode double haute résolution.	L
-16257	CØ7F	**CLRIOUDIS** ou **ENIOU**	Mise à Ø de IOUDIS pour autoriser l'accès des E/S sur l'espace CØ58-CØ5F.	E
-16256	CØ8Ø	**CØ8Ø**	Sélectionne la MEV de l'espace DØ-FF en lecture seulement et le banc 2 de l'espace DØ-DF.	L
-16255	CØ81		Sélectionne la MEM en lecture et la MEV entre DØ et FF en écriture et le banc n° 2 de l'espace DØ-DF.	LL
-16254	CØ82		Sélectionne la MEM et son banc n° 2 en lecture et empêche l'écriture en MEV.	L

Adresses Dec	Hex	Nom	Rôle	L = lire, E = écrire
-16253	CØ83		Autorise la lecture et l'écriture sur la MEV de l'espace DØ-FF et le banc n° 2.	LL
-16252 à -16249	CØ84 à CØ87		Effets identiques à ceux de CØ8Ø-CØ83.	
-16248	CØ88		Même effet que CØ8Ø sauf que le banc commuté en DØ-DF est le banc n° 1.	L
-16247	CØ89		Même effet que CØ81 mais banc n° 1.	LL
-16246	CØ8A		Même effet que CØ82 mais banc n° 1.	L
-16245	CØ8B		Même effet que CØ83 mais banc n° 1	LL
-16244 à -16241	CØ8C à CØ8F		Effets identiques à ceux de CØ88-CØ8D.	
-1624Ø à -16235	CØ9Ø à CØ9F	**DEV SEL 1**	16 adresses attribuées à l'interface - Série ou Port1.	
-16232	CØ98		Registre de réception/transmission dans l'ACIA du Port 1 (imprimante).	
-16231	CØ99		Registre d'état de l'ACIA du Port1.	
-1623Ø	CØ9A		Registre de commande de l'ACIA P1.	
-16231	CØ9B		Registre de contrôle de l'ACIA P1.	
-16234	CØAØ	**DEV SEL 2**	16 adresses attribuées à l'interface. Série 2 ou Port2.	
-16216	CØA8		Registre de réception/transmission de l'ACIA du Port2 (Modem).	
-16215	CØA9		Registre d'état de l'ACIA du Port2.	
-16214	CØAA		Registre de commande de l'ACIA P2.	
-16213	CØAB		Registre de contrôle de l'ACIA P2.	
-162Ø8 à -16193	CØBØ à CØBF	**DEV SEL 3**	16 adresses attribuées à l'interface Vidéo 8Ø colonnes.	
-16192 à -16177	CØCØ à CØCF	**DEV SEL 4**	16 adresses attribuées à l'interface Souris.	
-16176 à -16161	CØCØ à CØDF	**DEV SEL 5**	16 adresses attribuées à l'interface Souris.	
-1616Ø à -16145	CØEØ à CØEF	**DEV SEL 6**	16 adresses attribuées à l'interface des lecteurs de disquettes.	
-16144 à -16129	CØFØ à CØFF	**DEV SEL 7**	16 adresses attribuées à l'interface des lecteurs de disquettes.	

Adresses Dec	Adresses Hex	Nom	Rôle
-16128	C100	**SERSLOT**	Début du sous-programme de commande de l'imprimante connectée au **Port1**.
-15872	C200	**COMSLOT**	Début du sous-programme en MEM de commande au Modem connecté au **Port2**.
-15616	C300	**C3ENTRY**	Début du sous-programme en MEM de gestion de l'affichage en 80 colonnes.
-15599	C311	**MOVEAUX**	Entrée du sous-programme pour déplacer des zones de la mémoire vive à la mémoire auxiliaire et vice-versa.
-15596	C314	**XFER**	Entrée du programme qui transfère l'exécution sur la mémoire vive ou sur la mémoire auxiliaire.
-15593	C317	**BASICINIT**	Initialisation déclenchée par PR£3 de la gestion des 80 colonnes.
-15560	C338	**COPYROM**	Si la MEV D0-FF est sélectionnée en lecture, ce sous-programme y recopie le programme Monitor (F800-FFFF).
-15532	C354	**RESETLC**	Remet la MEV en écriture et autorise la lecture d'après ROMSTATE.
-15520	C360	**SETROM**	Met la MEM en lecture et la MEV en écriture et sauve RDLCRAM.
-15463	C399	**UPSHIFT0**	Met le bit 7 du caractère capté par NXTCHR à 1 si nécessaire.
-15450	C3A6	**GETCOUT**	Affichage de caractères normaux différents des caractères de contrôle.
	C3B8	**STORCH**	Après avoir saisi la position actuelle du curseur, l'affiche en tenant compte de INVFLAG.
-15360	C400		Début du sous-programme en MEM de prise en compte de la Souris.
-15332	C41C	**INITMOUSE**	Initialisation de la Souris.
-15299	C43D	**XSETMOUS**	Affectation du mode Souris d'après la valeur de l'accumulateur.
-15276	C454	**SETIOU**	Affectation du mode d'interruption d'après l'accumulateur.
-15251	C46D	**XMHOME**	Mise en position initiale et dans l'état initial de la Souris.
-15228	C484	**XMCLEAR**	Positionne la Souris en 0,0.
-15211	C495	**XMREAD**	Saisit les nouvelles valeurs.
-15184	C4B0	**XMCLAMP**	Enregistre les limites d'après A, min1,h et max1,h.
-15108	C4FC	**SERVEMOUSE**	Sous-programme de prise en charge d'interruptions demandées par la Souris.

ADRESSES ENTRE CØØØ et CFFF (zone d'E/S et des commutateurs)

Adresses Dec	Hex	*Nom*	*Rôle*
-14949	C59B	**HEXTODEC**	Convertit X,A en son équivalent décimal +ØØØØ et le range en $2ØØ,Y.
-14848	C6ØØ		Début du sous-programme en MEM de commande des lecteurs de disquette.
-14592	C7ØØ		Suite du précédent ; déclenche la mise en route à partir du lecteur externe de disquettes.
-14528	C74Ø	**TBL1**	Données du sous-programme EXERCICE.
-14519	C749	**TBL2**	Données du sous-programme EXERCICE.
-1451Ø	C752	**EXERCICE**	Programme pour s'exercer à la commutation des mémoires et des pages graphiques.
-14464	C78Ø	**XMBASIC**	Appel depuis un programme en Basic des fonctions de la Souris (PR£4).
-14433	C79F	**BASICIN**	Exécution du INPUT "";X,Y,S.
-1437Ø	C7DE	**MPADDLE**	Utilise la Souris en guise de manettes.
-14333	C8Ø3	**NEWIRQ**	Sous-programme principal de prise en charge d'interruptions.
-14174	C8A2	**MOVEIRQ**	Transfère le contenu de $FFFE,$FFFF (vecteur IRQ) en mémoire vive P et X.
-14139	C8C5	**XRDSER**	Sous-programme d'entrée Série.
-14Ø8Ø	C9ØØ	**ACIAINT**	Reconnaît si l'interruption provient d'un des ACIA ou du clavier.
-13889	C9BF	**XRDKBD**	Lit les tampons de l'entrée Série ou du clavier.
-13881	C9C7	**CMDTABLE**	Table des commandes au sous-programme Série appelée par $C1ØØ.
-13845	C9EB	**COMMAND**	Interprétation des commandes Série.
-135Ø5 à -13519	CB28 à CB2F	**COMTBL**	Valeurs standards de configuration des ports Série (largeur impression 8Ø colonnes, Modem, 3ØØ bauds, etc.).
-1352Ø	CB3Ø	**SCROLLDN**	Défilement vers le bas dans la fenêtre en 4Ø et en 8Ø colonnes.
-13515	CB35	**SCROLLUP**	Défilement vers le haut dans la fenêtre en 4Ø et en 8Ø colonnes.
-13374	CBC2	**DOCLR**	Exécute CLREOL en 4Ø ou en 8Ø colonnes.
	CCØ4	**CLRPORT**	Désactive le 'typeahead' et masque les interruptions externes.

Adresses Dec	Hex	*Nom*	*Rôle*
-13283	CC1D	**PICKY**	Retourne dans l'accumulateur le caractère actuellement sous le curseur en 4Ø ou en 8Ø colonnes pour répondre au caractère flèche droite.
-13236	CC4C	**SHOWCUR**	Affiche les différents types de curseur (damier clignotant, rectangle blanc ou le caractère-curseur) d'après la valeur de CURSOR ($7FB).
-132ØØ	CC7Ø	**UPDATE**	Incrémente RNDL,H et met à jour le curseur.
-13155	CC9D	**GETCURSOR**	Renvoie dans Y la valeur correcte de la position horizontale du curseur pour un STA (BASL),Y.
-13139	CCAD	**GETCUR2**	Sauvegarde dans Y le nouveau curseur.
-131Ø8	CCCC	**NEWESC**	Interprétation des caractères d'échappement (esc.) appelée par RDCHAR.
-13Ø75	CCED	**ESCRDKEY**	Exécute RDKEY mais avec les caractères d'échappement.
-12968	CD58	**CTLCHAR**	Interprétation des caractères de contrôle si les 8Ø colonnes sont actives, sinon retour à VIDOUT1.
	CDA5	**HOMECUR**	Exécute HOME.
-12851	CDCD	**CHK8Ø**	Sous-programme qui convertit en 4Ø colonnes si besoin est.
	CEØA	**WNDREST**	Restaure la fenêtre standard.
-1241Ø	CF86	**MOVEAUX**	Sous-programme de déplacement de zones entre les mémoires principale et auxiliaire ; Début→A1, Fin→A2, Dest→A4, C=1 P→X, C=Ø X→P.
-12339	CFCD	**XFER**	Sous-programme de transfert d'exécution entre Mev P et X ; Ad de transfert 3ED, C=1→X, C=Ø→P, pile et page zéro définies par V, V=1 P, V=Ø X.
-12289	CFFF	**CLRROM**	Désactivation des sous-programmes entre C8ØØ et CFFF.

Codes et adresses des commandes MONITOR

Commandes		*Adresses des sous-programmes*	
Code	*Nom*	*Nom*	*Adresse*
BC	'Ctrl C'	BASCONT	FEB3
B2	'Ctrl Y'	USR	FECA
BE	'Ctrl E'	REGZ	FEBF
EF	V	VERIFY	FE36
C4	'Ctrl K'	INPRT	FE8D
A9	'Ctrl P'	OUTPRT	FE97
BB	'Ctrl B'	XBASIC	FEBØ
A6	-	SETMODE	FE18
A4	+	SETMODE	FE18
Ø6	M	MOVE	FE2C
95	<	LT	FE2Ø
Ø7	N	SETNORM	FE84
Ø2	I	SETINV	FE8Ø
Ø5	L	LIST	FE5E
ØØ	G	GO	FEB6
93	:	SETMODE	FE18
A7	.	SETMODE	FE18
C6	'Return'	CRMON	FEF6
99	espace	BLANK	FFØ4

Le rôle des commandes est expliqué dans le chapitre "**Commandes**".

La colonne code est dérivée du code ASCII par la formule :

```
EOR £BØ
ADC £88 (C=1)
```

L'ordre de ce tableau est celui de la table des commandes CHRTBL d'adresses FFCC-FFE2. La table des vecteurs de commandes est SUBTBL d'adresses FFE3-FFF9 et ne contient que la partie basse des adresses -1 ; la partie haute est égale à FE.

Les grandes fonctions du MONITOR

1 - Début

-155	FF65	**MON**	Entrée avec un 'bip' dans le Monitor.
-151	FF69	**MONZ**	Entrée silencieuse dans le Monitor.

2 - Entrée de données (seront placées en $200 sur X caractères)

-665	FD67	**GETLNZ**	Saisie d'une ligne de commande avec affichage du caractère de sollicitation.
-651	FD75	**NXTCHAR**	Lecture du prochain caractère à l'aide de ESCRDKEY.
-715	FD35	**RDCHAR**	Lecture d'un caractère à l'aide de ESCRDKEY.
-13075	CCED	**ESCRDKEY**	Lit une touche éventuellement 'esc' en appelant RDKEY puis NOESCAPE.
-756	FD75	**RDKEY**	Fait afficher le bon curseur, attend qu'une touche soit tapée, détecte 'esc' ou flèche droite.
-13108	CCCC	**NEWESC**	Appelée par RDKEY, gère le mode d'échappement.
-700	FD44	**NOESCAPE**	Annule le mode d'échappement et autorise les caractères de contrôle.
-707	FD3D	**NOTCR**	Appelle GETCOUT puis détecte 'Ctrl X' et la flèche gauche.
-15450	C3A6	**GETCOUT**	Affiche le caractère saisi au clavier mais filtre les caractères de contrôle.
-636	FD84	**ADDINP**	Enregistrement de l'accumulateur dans la Xème position du tampon d'entrée $200 jusqu'à un 'Return'.

3 - Analyse des données et interprétation

-144	FF70		Analyse du tampon $200 avec MODE=0.
-141	FF73	**NXTITM**	Analyse de l'item suivant.
-89	FFA7	**GETNUM**	Décode un nombre dans le tampon.
-83	FFAD	**NXTCHR**	Analyse le prochain caractère, met à 1 le bit 7 si nécessaire.
-13	FFF3	**GETHEX**	Récupère un chiffre hexadécimal à l'aide de DIG et le met en A2L,H sous son code ASCII.
-134	FF7A	**CHRSRCH**	Appelé par NXTITM pour reconnaître la commande et appelle TOSUB et renvoie à NXTITM.
-66	FFBE	**TOSUB**	Appel du sous-programme correspondant à la commande en empilant les poids forts et faibles à l'adresse du sous-programme et en mettant MODE à 0.
-132	FFC7	**ZMODE**	Met à 0 MODE.
-488	FE18	**SETMODE**	Attribution du mode d'après le caractère saisi, - + . :

4 - Affichage des registres

-55Ø	FDDA	**PRBYTE**	Affiche (A) en deux chiffres hexadécimaux.
-541	FDE3	**PRHEX**	Le chiffre des poids faibles de (A).
-1728	F94Ø	**PRNTYX**	(Y) (X) en quatre chiffres hexa.
-1727	F941	**PRNTAX**	(A) (X) en quatre chiffres hexa.
-1724	F944	**PRNTX**	(X) en deux chiffres hexa.
-321	FEBF	**REGZ**	Appel REGDSP pour afficher tous les registres.
-1321	FAD7	**RGDSP**	Les noms et les valeurs des registres sont affichés.

5 - Sortie des caractères (sur écran ou non)

-626	FD8E	**CROUT**	Retour chariot donc retour à la ligne suivante.
-531	FDED	**COUT**	Saut au début du sous-programme de sortie à l'aide du vecteur CSWL,H.
-528	FDFØ	**COUT1**	Début du sous-programme avec test de l'activation des 8Ø colonnes à l'aide des indicateurs de VFACT ($67B) et saut à DOCOUT1.
-11ØØ	FBB4	**DOCOUT1**	Applique AND INVFLAG si le caractère n'est pas un caractère de contrôle et envoie sur COUTZ.
-522	FDF6	**COUTZ**	Affiche le caractère de A mais avec sauvegarde de Y et A au cas d'une suspension d'affichage due à 'Ctrl S'.
-116Ø	FB78	**VIDWAIT**	Teste 'Ctrl S' pendant l'affichage et assure la reprise puis teste VFACT ; si 4Ø colonnes, alors VIDOUT ; si caractère de contrôle, alors DOCTL ; sinon fait NEWADV.
-1196	FB54	**DOCTL**	Met le bit 7 à 1 et fait exécuter l'ordre associé au caractère de contrôle par CTLCHARØ.
-12972	CD54	**CTLCHARØ**	Traitement des caractères de contrôle quand la gestion 8Ø colonnes est active.
-15432	C3B8	**STORCH**	Détermine la position du curseur avec GETCUR, applique INVFLG et affiche.
-1Ø27	FBFD	**VIDOUT**	Affichage du caractère en 4Ø colonnes sans la gestion 8Ø colonnes en ligne avec STORADV et VIDOUT1.
-1Ø4Ø	FBFØ	**STORADV**	Avance le curseur en contrôlant la fenêtre et fait afficher par VIDOUT1.
-1Ø2Ø	FCØ4	**VIDOUT1**	Affichage prenant en compte les caractères Ctrl M, Ctrl J, Ctrl H, Ctrl G.
-112Ø	FBAØ	**NEWADV**	Avance le curseur et met à jour OLDCH et CH si 4Ø colonnes mais sous contrôle des 8Ø colonnes.

-198	FF3A	**BELL**	Emet un joli 'bip'.
-1063	FBDF	**BELL1**	Retarde le 'bip'.
-1052	FBE4	**BELL2**	Stimule le haut-parleur à 1Khz pendant 0.1 seconde.
-1720	F948	**PRBLNK**	Laisse trois espaces.
-384	FE80	**SETINV**	Met $3F dans INVFLG.
-380	FE84	**SETNORM**	Met $FF dans INVFLG.
-12870	CDB0	**X.S0**	Exécute l'ordre 'Ctrl N' d'affichage en mode normal.
-12863	CDB7	**X.SI**	Exécute l'ordre 'Ctrl O' d'affichage en mode inversé.
-12887	CD9F	**MOUSE.OFF**	Empêche l'affichage des icones, ordre donné par PRINT CHR$(24).
-12903	CD99	**MOUSE.ON**	Autorise l'affichage des icones, l'ordre est PRINTCHR$(27).

6 - Gestion du curseur

-13155	CC9D	**GETCUR**	Renvoie dans Y la position horizontale du curseur si la gestion 80 colonnes est active.
-1104	FBB0	**NEWADV1**	Avancée du curseur si 80 colonnes.
-1036	FBF4	**ADVANCE**	Avancée du curseur si 40 colonnes.
-926	FC62	**CR**	Saut à la ligne avec NEWCR.
-909	FC73	**NEWCR**	Met la position horizontale CH à 0 et fait un saut de ligne si le programme courant est en Basic.
-922	FC66	**LF**	Saut de ligne.
-1008	FC10	**BS**	Recul d'une position à l'aide de DECCH.
-286	FEE2	**DECCH**	Diminue CH de 1.
-279	FEE9	**CLRCH**	Met CH à 0.
-276	FEEC	**SETCUR**	Donne à CH la valeur de A.
-998	FC1A	**UP**	Remontée éventuelle du curseur.
-990	FC22	**VTAB**	Avec un nouveau CV, il faut recalculer BASH,L en appelant BASCALC.
-1087	FBC1	**BASCALC**	Calcul de l'adresse de base, c'est-à-dire de l'octet le plus à gauche de la ligne désignée dans A.

7 - Gestion de l'écran-texte

-12773	CE1B	**HOOKITUP**	Activation des sous-programmes et des commutateurs pour la gestion des 80 colonnes.
-12866	CDBE	**SET80**	Affichage sur 80 colonnes (esc 8).
-12864	CDC0	**SET40**	Affichage sur 40 colonnes (esc 4).
-12851	CDCD	**CHK80**	Conversion de 80 en 40 colonnes.
-1223	FB39	**SETTEXT**	Met l'écran en mode texte et la fenêtre standard.
-1205	FB4B	**SETWND**	Met en place la fenêtre standard.

-12756	CEØA	**WNDRST**	Restaure la fenêtre standard.
-936	FC58	**HOME**	Saut à HOMECUR et CLREOP1.
-12891	CDA5	**HOMECUR**	Donne la valeur Ø à la position horizontale du curseur et WNDTOP à CV ; recalcule BASL,H.
-931	FC5D	**CLREOP1**	Donne à Y la valeur juste de CH et fait effacer toute la fenêtre.
-958	FC42	**CLREOP**	Effacement jusqu'au bas de la fenêtre.
-868	FC9C	**CLREOL**	Saut à NEWCLREOL pour effacer jusqu'en bout de ligne.
-883	FC8D	**NEWCLREOL**	Recherche la position du curseur avant d'afficher des espaces sur le reste de la ligne à l'aide de DOCLR.
-864	FCAØ	**CLRLIN**	Effacement de toute la ligne courante.
-13374	CBC2	**DOCLR**	Exécution de l'effacement de ligne.
-912	FC7Ø	**SCROLL**	Saut à SCROLLUP pour un décalage de l'image d'une ligne vers le haut.
-13515	CB35	**SCROLLUP**	Exécute le décalage vers le haut en 4Ø ou en 8Ø colonnes.
-1352Ø	CB3Ø	**SCROLLDN**	Exécute un décalage d'une ligne vers le bas de toute l'image de l'écran.
-12737	CE45	**QUIT**	Désactivation de la gestion des 8Ø colonnes et restauration des vecteurs d'E/S pour 4Ø colonnes KEYIN et COUT1.

8 - Graphique basse-résolution

-1216	FB4Ø	**SETGR**	Mode mixte et basse résolution avec effacement noir.
-1948	F864	**SETCOL**	Affectation d'une couleur d'après l'accumulateur COLOR=17*A MOD 16.
-2Ø48	F8ØØ	**PLOT**	Voir détails page 126.
-2Ø34	F8ØE	**PLOT1**	
-2Ø23	F819	**HLINE**	
-2Ø2Ø	F81C	**HLINE1**	
-2Ø1Ø	F826	**VLINEZ**	
-2ØØ8	F828	**VLINE**	
-1998	F832	**CLRSCR**	
-1994	F836	**CLRTOP**	
-1977	F847	**GBASCALC**	Calcule l'adresse du premier octet de la ligne graphique spécifiée dans A, la range en GBASH,L.
-1935	F871	**SCRN**	Met dans A le code de la couleur du domino situé en Y,A.

9 - Entrée-sortie

-125Ø	FB1E	**PREAD**	Lecture des manettes ou des coordonnées de la Souris, Ø ou 1 dans X et le résultat dans Y.

-375	FE89	**SETKBD**	Met le clavier comme dispositif d'entrée (IN£Ø).
-371	FE8D	**INPRT**	Met en ligne le dispositif dont le numéro d'interface vaut X (IN£s).
-365	FE93	**SETVID**	Met l'écran vidéo en ligne (PR£Ø).
-361	FE97	**OUTPRT**	Met en ligne en sortie le dispositif dont le numéro d'interface vaut X (PR£s).
-318	FEC2	**OPRTØ**	Exécute PR£Ø, met le curseur en damier et appelle DOPRO.
-3Ø6	FECE	**DOPRØ**	Prédispose le jeu primaire des caractères, appelle CHK8Ø et conditionne l'affichage traditionnel COUT1.
-15332	C41C	**INITMOUSE**	Initialise la prise en compte des déplacement de la Souris.
-133Ø8	CCØ4	**CLRPORT**	Inhibe les interruptions externes et le "typeahead".

1Ø - Désassemblage

-418	FE5E	**LIST**	Exécute l'ordre L de désassemblage de 2Ø instructions.
-413	FE63	**LIST2**	Désassemble (A) instructions.
-184Ø	F8DØ	**INSTDSP**	Désassemble une instruction dont l'adresse est en Y,X.
-1918	F882	**INSDSP1**	Affiche son adresse d'implantation.
-19Ø6	F88C	**INSDSP2**	Détermine son code-opération.
-1879	F8A9	**GETFMT**	Recherche le nombre d'octets de l'instruction d'après son code et décrypte le code.
-17Ø9	F953	**PCADJ**	Ajuste la valeur du compteur ordinal.

11 - Affichage des mémoires

-589	FDB3	**XAM**	Affiche les contenus des positions de mémoire pointées par A1L,H jusqu'à (A2).
-622	FD92	**PRA1**	Affiche l'adresse et le tiret puis :
-586	FD86	**DATAOUT**	Affiche les contenus.
-838	FCBA	**NXTA1**	Incrémente A1L,H jusqu'à (A2L,H).

12 - Déplacement de mémoires et comparaison de zones

-468	FE2C	**MOVE**	Déplacement de (A1)-(A2) vers (A4).
-844	FCB4	**NXTA4**	Incrémente A4L,H et A1L,H.
-458	FE36	**VFY**	Comparaison des zones (A1)-(A2) et (A4) et affichage des différences.
-48Ø	FE2Ø	**LT**	Transfert de A2 en A5 et A5.
-1241Ø	CF86	**MOVEAUX**	Déplacement vers ou de la MEV X depuis (A1) jusqu'à (A2) vers (A4) ; C=1 de P à X, C=Ø de X à P.

-14174	C8A2	**MOVEIRQ**	Transfère les contenus du vecteur d'interruption figés en MEM $FFFF, $FFFE dans les mémoires P et X.
-1556Ø	C338	**COPYROM**	Recopie en MEV aux mêmes adresses le programme Monitor ($F8ØØ-FFFF) ; il faudra autoriser la lecture de la MEV.

13 - Arithmétique hexadécimale

-57Ø	FDC6	**XAMPM**	Addition ou soustraction de A1L et A2L avec affichage du résultat précédé de =
-14949	C59B	**HEXTODEC**	Convertit X,A, 16 bits en Y chiffres décimaux rangés en $2ØØ précédés du signe et suivis d'une virgule.

14 - Système superviseur

-14333	C8Ø3	**NEWIRQ**	Prise-en-charge d'interruptions avec sauvegarde de l'accumulateur sur la pile et aussi de l'état de commutation des mémoires. Reconnaît l'interruption venant soit de BRK (saut à NEWBRK), soit de la Souris (saut à MOUSEINT) ou des périphériques. Série (saut à ACIAINT) sinon saute à ($3FE).
-142Ø6	C882	**IRQDONE**	Restauration de l'état de la machine après résolution d'une interruption.
-1465	FA47	**NEWBRK**	Si une instruction 'BRK' a eu lieu, alors l'état de la machine est rangé en MACSTAT et les registres sont restaurés avant d'appeler BREAK.
-146Ø	FA4C	**BREAK**	Sauvegarde de P, A, X, Y, PC et saut à ($3FØ), normalement FA59.
-1447	FA59	**OLDBRK**	Affiche les registres et le PC et renvoie à MON,$FF65.
-1438	FA62	**RESET**	Sous-programme de redémarrage déclenché avec les touches 'Ctrl Reset'.
-1233	FB2F	**INIT**	Impose un état initial déterminé (écran-fenêtre-souris-port-clavier) et appelle RESET.X puis NEWMON.
-14Ø7	FA81	**NEWMON**	Détermine le type de redémarrage d'après la valeur de $3F4,PWREDUP (si $3F4=3F3+$A5 alors à chaud, vers l'adresse rangée en $3F2, $3F3, SOFTEV).
-1347	FABD	**RESET.X**	Initialise le mode d'affichage VMODE à $FF ; teste si les deux touches pommes sont enfoncées pour déclencher le programme EXERCICE, sinon si la touche "pomme ouverte ou blanche" est enfoncée, alors redémarrage à froid à l'aide de PWRUP.

-137Ø	FFA6	**PWRUP**	Appelle COLDSTART puis fait SETPG3.
-1367	FAA9	**SETPG3**	Charge les vecteurs de la page 3 avec les valeurs standard ; 3F2-3F4 (Basic) avec ØØ EØ 45 et 3FØ-3F1 avec 59 FA (OLDBRK) puis fait afficher Apple II C et saute à $C6ØØ.
-822	FCCA	**COLDSTART**	Attention, les pages BF à Ø2 sont touchées : deux positions variables de mémoire dans chaque page sont effacées (dans la page 3, c'est le vecteur RESET). Les valeurs standards de dimensionnement des périphériques Série figées entre CB27 et CB7F devraient être placées dans les mémoires 4F8 à 4FF en MEV auxiliaires (la partie adresse de l'instruction en FCE7 n'est-elle pas fausse ?).
-1184	FB6Ø	**APPLEII**	Affiche 'Apple II C' en haut de l'écran.
-1169	FB6F	**SETPWRC**	Fait l'opération EOR #$A5 avec $3F3 et range le résultat en $3F4, PWEREDUP.
-856	FCAB 8	**WAIT**	Temporisation égale à 13+2712*A+512*A*A.
-336	FEBØ	**XBASIC**	Démarrage à froid du Basic (saut en $EØØØ).
-333	FE83	**BASCONT**	Continuation de l'interprétation Basic (saut à $EØØ3).
-33Ø	FEB6	**GØ**	Exécution de l'ordre 'G'.
-266	FEF6	**CRMON**	Interprète l'ordre 'Return' comme le caractère d'espacement, s'il est la seule commande au Monitor.
-31Ø	FECA	**USR**	Exécute l'ordre 'Ctrl Y' en sautant à l'adresse $3F8.
-193	FF3F	**RESTORE**	Restaure A,X,Y,S,P en provenance de $45,$46,$47,$48 et du haut de la pile.
-182	FF4A	**SAVE**	Sauvegarde A,X,Y,P,S en $45,$46,$47,$48,$49.
-211	FF2D	**PRERR**	Affiche ERR et émet un bip.
	FBBF	ROMVER	Affiche la version de la MEM

Commandes monitor

Appel BASIC	*Appel* Monitor	*Nom*	*Résultat*
CALL-2048	F8ØØG	PLOT	Y, A — calcul de GBASL,H
CALL-2034	F8ØEG	PLOT1	Y — à l'ordonnée courante
CALL-2023	F819G	HLINE	Y, A, ($2C) — A et Y sont modifiés
CALL-2020	F81CG	HLINE1	Y, ($2C) — à la ligne courante
CALL-2010	F826G	VLINEZ	Y, (A)+1+(C), ($2D) — A est modifié
CALL-2008	F828G	VLINE	Y, A, ($2D)
CALL-1998	F832G	CLRSCR	effacement de l'écran graphique et mise à @ du texte
CALL-1994	F836G	CLRTOP	effacement de l'écran graphique. Les 4 lignes du bas ne sont pas touchées
CALL-1992	F838G	CLRSC2	Y — effacement jusqu'à la ligne Y
CALL-1988	F83CG	CLRSC3	Y, ($2D) — effacement de l'écran dans la partie haute et à gauche

SOFTEV ET PWREDUP $3F2, $3F3 ET $3F4

Le Monitor en MEM a de quoi déterminer son type de redémarrage en cas de 'Reset'.

Le 'Reset' à chaud est programmable dans le vecteur SOFTEV et le monitor se sert d'un octet particulier PWREDUP comme trace du passage par un démarrage à froid. PWREDUP doit être le ou exclusif # $A5 et du contenu de $3F3 si on veut que 'Reset' produise un démarrage à chaud.

Valeur par défaut :

	SOFTEV	PWREDUP	PWREDUP	SOFTEV
sans SED	$EØØ3	$45 (Basic à chaud)		$EØØ3
avec SED	$9DBF	$38	$1B	$BEØØ PRODOS

Si au démarrage à froid, dès la mise sous tension, une carte d'interface de lecteur de disquette est reconnue, alors le monitor laisse s'exécuter le programme de chargement du SED (Bootstrap) présent en MEM sur la carte d'interface.

Dans l'ensemble des fonctions de mise en route du SED, l'affectation d'une adresse de redémarrage à chaud (en cas de 'Reset') est réalisée dans SOFTEV ($9DBF) (poids faible en tête).

SOFTEV peut être modifié par l'utilisateur qui aura soin d'affecter à PWREDUP le ou exclusif de SOFTEV+1 et de $A5 pour que 'Reset' conduise à un programme spécifique et ne provoque pas l'équivalent d'un démarrage à froid.

AMPERV $3F5, $3F6, $3F7

L'adresse AMPERV sera utilisée (par indirection) pour déclencher l'exécution d'un programme en langage machine depuis un programme en APPLESOFT contenant le MOT-CLE &.

Exemple : * 3F5 : 4C ØØ Ø3 JMP $3ØØ

Le mot-clé & conduira à l'exécution du sous-programme commençant à l'adresse $3ØØ.

DOSWARMSTART $3DØ : 4C BF 9D

Cette adresse est à utiliser pour reprendre l'APPLESOFT (contrôlé par le DOS 3.3) depuis le Monitor, à savoir :

```
* 3DØG
]
```

Le programme courant n'a pas été touché.

MONZ $FF69 (-151)

Adresse d'entrée dans le Monitor.
Depuis l'APPLESOFT, on tape

CALL - 151

alors *

Routines fondamentales

Impression d'un caractère

COUT	$FDED	Affiche le caractère présent dans l'accumulateur et avance le curseur. Prend en compte 'Return' 'LF' et les 2 modes Normal/Inverse.
OUTDO	$DB5C	Dans APPLESOFT, affiche le caractère présent dans l'accumulateur et prend en compte les 3 modes video Normal/Inverse/Clignotant.

Acquisition d'un caractère dans le texte BASIC

CHRGET	$ØØB1	Ce sous-programme (qui s'auto-modifie) pointe en $B8, $B9 le caractère à prendre qui sera chargé dans l'accumulateur Z = 1 si fin d'instruction ($3A ou $ØØ) C = Ø si le caractère est un chiffre (les espaces du texte en BASIC ont été sautés).
CHRGOT	$ØØB7	Le caractère est le caractère actuel non le suivant comme dans CHRGET.

Lecture d'un caractère tapé au clavier

RDKEY	$FDØC	Attend qu'une touche soit pressée avec le curseur clignotant. Le code du caractère est chargé dans l'accumulateur

Nom	Adresse HEX	Adresse DEC	Rôle
TXTTAB	$67,$68	103,104	Début du texte BASIC = $8Ø1 (2049) par défaut.
VARTAB	$69,$6A	105,106	Début des variables simples, des pointeurs de chaînes, des pointeurs de fonction.
ARYTAB	$6B,$6C	107,108	Début des variables dimensionnées, pointeurs des tableaux de chaînes.
STREND	$6D,$6E	109,110	Début de l'espace libre.
FRETOP	$6F,$7Ø	111,112	Fin de l'espace libre. Fin des chaînes.
MEMSIZ	$73,$74	115,116	Début des chaînes. Fin de l'espace mémoire +1. Les chaînes sont enregistrées du haut vers le bas.
CURLIN	$75,$76	117,118	Numéro de la ligne en cours d'exécution.
OLDLIN	$77,$78	119,120	N° de la ligne interrompue par 'Ctrl C', STOP ou END.
OLDTXT	$79,7A	121,122	Adresse du dernier octet (ØØ) de la ligne en cours d'exécution.
DATLIN	$7B,$7C	123,124	N° de la ligne dans laquelle sont lues les DATA.
DATPTR	$7D,$7E	125,126	Adresse du premier octet des DATA à lire.
INPPTR	$7F,$8Ø	127,128	Pointe le tampon d'entrée par clavier pendant INPUT.
VARNAM	$81,$82	129,130	Contient le nom (2 caractères) de la dernière variable référencée.
VARPNT	$83,84	131,132	Adresse de la valeur de la dernière variable référencée, ou de l'octet de longueur d'une chaîne.
PGEND	$AF,$BØ	175,176	Fin du texte BASIC.

Résumé

HIMEM:		MEMSIZ
	chaînes	FRETOP
	libre	STREND
	tableaux	ARYTAB
	variables	VARTAB
LOMEM: $8Ø1	texte	PGEND
		TXTTAB

Implantation d'un programme et des variables en MEV

```
*
]LIST

10 AA = 2
20 AA% = 4
30 AA$ = ""
40  DIM AA(1,3)
50  DIM AA%(2,1)
60  DIM AA$(3,2)
70  DEF FN AA(X)= X - 256 * INT
     (X / 256)
80  PRINT  FN AA(257)

]RUN
1

]CALL -151

*69.6A

0069- 74 08
*800.873
```

AA variable réelle simple
AA% variable entière simple
AA$ variable chaîne de caractère simple

AA(1,3) variable réelle dimensionnée
AA%(2,1) variable entière dimensionnée
AA$(3,2) variable chaîne dimensionnée

FN AA(X) fonction définie par le programme

```
0800- 00<0A 08>[0A 00]41 41(D0
0808- 32 00<14 08>[14 00] 41 41
0810- 25 (D0 34 00<1F 08>[1E 00]
0818- 41 41 24 (D0 22 22 00<2C
0820- 08>[28 00](86 41 41 (28 31
0828-(2C 33 (29 00<3A 08>[32 00]
0830-(86 41 41 (25 28 32 2C 31
0838-(29 00<48 08>[3C 00](86 41
0840- 41 24 (28 33 2C 32 (29 00
0848-<63 08>[46 00](B8( C2 41 41
0850-(28 58 (29 (D0 58( C9 32 35
0858- 36 (CA (D3 (28 58 (CB 32 35
0860- 36 (29 00<71 08>[50 00](BA
0868- (C2 41 41 (28 32 35 37(29
0870- 00 00 00 0A
```

Texte Basic codé

< >adresse du début de la ligne d'instruction suivante

[ ] numéro de ligne d'instruction

(mot-clé

ØØ fin de ligne

ØØ ØØ fin du texte-programme

—— définition de fonction

Variables simples

```
*6B.6C

006B- 97 08
*874.896

0874-.41 41;82 00
0878- 00 00 00.C1 C1;00 04 00
0880- 00 00.41 C1;00,1D 08 00
0888- 00.C1 41;54 08,92 08,58.
0890-.58 00;00 00 00 00 00.
```

2 caractères du nom ; valeur.

Codage des 2 caractères du nom :

		1er car.	2e car.
AA	réelle	P	P
AA%	entière	N	N
AA$	chaîne	P	N
AA(X)	fonction	N	P

P est le code ASCII avec le bit 7=Ø
N est le code ASCII avec le bit 7=1

Valeur d'une variable sur 5 octets :

	1er octet	2e octet	3e octet	4e octet	5e octet
Réelle	exposant	mantisse			
Entière	poids forts	poids faibles	non utilisés		
Chaîne	longueur	adresse du début de chaîne		non utilisés	
Fonction	adresse de la définition		adresse de l'argument		code P du 1er car. après =

Les adresses sont exprimées avec les poids faibles sur le 1er octet et les poids forts sur le 2ème octet.

Implantation des variables dimensionnées

```
*6D.6E

006D- 0A 09
*897.909

0897-.41                                  DIM AA(1,3)
0898- 41;<31 00>02 00 04 00 02
08A0-/00 00 00 00 00/00 00 00
08A8- 00 00/00 00 00 00 00/00
08B0- 00 00 00 00/00 00 00 00
08B8- 00/00 00 00 00 00/00 00
08C0- 00 00 00/00 00 00 00 00
08C8- C1 C1;<15 00>02 00 02 00            DIM AA%(2,1)
08D0- 03/00 00/00 00/00 00/00
08D8- 00/00 00/00 00/41 C1;<2D            DIM AA$(3,2)
08E0- 00>02 00 03 00 04 00 00
08E8- 00/00 00 00/00 00 00/00
08F0- 00 00/00 00 00/00 00 00/
08F8- 00 00 00/00 00 00/00 00
0900- 00/00 00 00/00 00 00/00
0908- 00 00/
```

. 2 caractères du nom ; offset / variable suivante

D nombre de dimensions ou d'indices

└──┘ nombre maximum d'éléments du tableau pour chaque dimension de la dernière à la première (valeur maximum de l'indice +1).

/ / valeur de chaque élément du tableau
- réelle 5 octets
- entière 2 octets seulement
- chaîne 3 octets : longueur, adresse

dans l'ordre AA(Ø,Ø), AA(1,Ø), AA(Ø,1), AA(1,1), AA(Ø,2), AA(1,2), AA(Ø,3), AA(1,3) l'indice le plus à droite augmentant le plus lentement.

Exemple d'implantation d'un programme et des variables en mémoire vive

```
]LIST

10  REM  APPLESOFT
20  INPUT "NOM ?";N$
30  FOR K = 1 TO  LEN (N$) - 2
40  PRINT  RIGHT$ (N$, LEN (N$) -
     K + 1);" "; MID$ (N$,K,2);"
     ";
50  NEXT K

]CALL -151
```

```
*69.6A

0069- 63 08
```

VARTAB

```
*67.68

0067- 01
0068- 08
```

TXTTAB

```
*800.862

0800- 00<11 08>[0A 00](B2 20 41
0808- 50 50 4C 45 53 4F 46 54
0810- 00<21 08>[14 00](84 22 4E
0818- 4F 4D 20 3F 22 3B 4E 24
0820- 00<32 08>[1E 00](81 4B D0
0828- 31 C1 E3 28 4E 24 29 C9
0830- 32 00<5A 08>[28 00](BA (E9
0838- 28 4E 24 2C E3 28 4E 24
0840- 29 C9 4B C8 31 29 3B 22
0848- 20 22 3B (EA 28 4E 24 2C
0850- 4B 2C 32 29 3B 22 20 22
0858- 3B 00<61 08>[32 00](82 4B
0860- 00 00 00
```

Texte BASIC codé

(mots-clés

ØØ fin de ligne

ØØ ØØ fin de programme

[ ] n° de ligne

< >adresse de la ligne suivante

Exemple d'implantation d'un programme et de ses variables en mémoire vive.
(suite)

]RUN NOM ?DUAERB DUAERB DU UAERB UA AERB AE ERB ER	**Exécution** des instructions
]CALL -151	
*6D.6E 006D- 71 08	STREND
*863.870 0863-.4E 80;06,FA 95/ 0868- 00 00.4B 00;83 20 00 00 0870- 00/	**Variables** -N$: longueur, pointeur - K : valeur réelle
*6F.70 006F- FA 0070- 95	FRETOP
*95FA.95FF 95FA- 44 55 41 45 52 42	Valeur de N$
*79.7A 0079- 60 08	adresse du dernier octet de la dernière instruction exécutée
*AF.B0 00AF- 63 00B0- 08	fin du programme (+2)

Les mots-clés par ordre croissant des codes et les adresses des sous-programmes correspondants dans l'interpréteur.

Codes $80 à $A9

Mot-clé	*Code Hexadécimal*	*Adresse Hexadécimale*
END	$80 128	$D870 55408
FOR	$81 129	$D766 55142
NEXT	$82 130	$DCF9 56569
DATA	$83 131	$D995 55701
INPUT	$84 132	$DBB2 56242
DEL	$85 133	$F331 62257
DIM	$86 134	$DFD9 57305
READ	$87 135	$DBE2 56290
GR	$88 136	$F390 62352
TEXT	$89 137	$F399 62361
PR£	$8A 138	$F1E5 61925
IN£	$8B 139	$F1DE 61918
CALL	$8C 140	$F1D5 61909
PLOT	$8D 141	$F225 61989
HLIN	$8E 142	$F232 62002
VLIN	$8F 143	$F241 62017
HGR2	$90 144	$F3D8 62424
HGR	$91 145	$F3E2 62434
HCOLOR=	$92 146	$F6E9 63209
HPLOT	$93 147	$F6FE 63230
DRAW	$94 148	$F769 63337
XDRAW	$95 149	$F76F 63343
HTAB	$96 150	$F7E7 63463
HOME	$97 151	$FC58 64600
ROT=	$98 152	$F721 63265
SCALE=	$99 153	$F727 63271
SHLOAD	$9A 154	$03F5 1013
TRACE	$9B 155	$F26D 62061
NOTRACE	$9C 156	$F26F 62063
NORMAL	$9D 157	$F273 62067
INVERSE	$9E 158	$F277 62071
FLASH	$9F 159	$F280 62080
COLOR=	$A0 160	$F24F 62031
POP	$A1 161	$D96B 55659
VTAB	$A2 162	$F256 62038
HIMEM:	$A3 163	$F286 62086
LOMEM:	$A4 164	$F2A6 62118
ONERR	$A5 165	$F2CB 62155
RESUME	$A6 166	$F318 62232
RECALL	$A7 167	$03F5 1013
STORE	$A8 168	$03F5 1013
SPEED=	$A9 169	$F262 62050

CODES ET ADRESSES DES MOTS-CLES APPLESOFT

Les mots-clés par ordre croissant des codes.

Codes $AA à $D4

Mot-clé	*Code Hexadécimal*	*Adresse Hexadécimale*
LET	$AA 170	$DA46 55878
GOTO	$AB 171	$D93E 55614
RUN	$AC 172	$D912 55570
IF	$AD 173	$D9C9 55753
RESTORE	$AE 174	$D849 55369
&	$AF 175	$03F5 1013
GOSUB	$B0 176	$D921 55585
RETURN	$B1 177	$D96B 55659
REM	$B2 178	$D9DC 55772
STOP	$B3 179	$D86E 55406
ON	$B4 180	$D9EC 55788
WAIT	$B5 181	$E784 59268
LOAD	$B6 182	$03F5 1013
SAVE	$B7 183	$03F5 1013
DEF	$B8 184	$E313 58131
POKE	$B9 185	$E77B 59259
PRINT	$BA 186	$DAD5 56021
CONT	$BB 187	$D896 55446
LIST	$BC 188	$D6A5 54949
CLEAR	$BD 189	$D66A 54890
GET	$BE 190	$DBA0 56224
NEW	$BF 191	$D649 54857
TAB(	$C0 192	
TO	$C1 193	
FN	$C2 194	
SPC(	$C3 195	
THEN	$C4 196	
AT	$C5 197	$DE98 56984
NOT	$C6 198	
STEP	$C7 199	$E7BE 59326
+	$C8 200	$E7A7 59303
-	$C9 201	$E97F 59775
*	$CA 202	$EA66 60006
/	$CB 203	$EF09 61187
^	$CC 204	$DF55 51173
AND	$CD 205	$DF4F 57167
OR	$CE 206	
>	$CF 207	$DF6A 57194
=	$D0 208	
<	$D1 209	
SGN	$D2 210	$EB90 60304
INT	$D3 211	$EC23 60451
ABS	$D4 212	$EBAF 60335

Les mots-clés par ordre croissant des codes.

Codes $D5 à $EA

Mot-clé	*Code Hexadécimal*	*Adresse Hexadécimale*
USR	$D5 213	$000A 10
FRE	$D6 214	$E2DE 58078
SCRN(	$D7 215	$D412 54290
PDL	$D8 216	$DFCD 57293
POS	$D9 217	$E2FF 58111
SQR	$DA 218	$EE8D 61069
RND	$DB 219	$EFAE 61358
LOG	$DC 220	$E941 59713
EXP	$DD 221	$EF09 61193
COS	$DE 222	$EFEA 61418
SIN	$DF 223	$EFF1 61425
TAN	$E0 224	$F03A 61498
ATN	$E1 225	$F09E 61598
PEEK	$E2 226	$E764 59236
LEN	$E3 227	$E6D6 59094
STR$	$E4 228	$E3C5 58309
VAL	$E5 229	$E707 59143
ASC	$E6 230	$E6E5 59109
CHR$	$E7 231	$E646 58950
LEFT$	$E8 232	$E65A 58970
RIGHT$	$E9 233	$E686 59014
MID$	$EA 234	$E691 59025

Classement par grandes fonctions

1 - Début (points d'entrée)

-8192	EØØØ	BASIC	démarrage à froid si 'Crtl B'.
-3800	F128	COLDSTART	démarrage à froid si 'Ctrl B'.
-8189	EØØ3	BASIC2	2ème entrée dite à chaud si 'Ctrl C' ou 'Reset'.
-11204	D43C	CMDLOOP	début de la bouche principale de l'interpréteur (à chaud).

2 - Entrée des données (dans le tampon $2ØØ)

-11201	D43F	entrée avec	affichage de]
-10962	D52E	INLIN+2	entrée par le clavier d'une instruction avec (X) comme signal (PROMPT)
-10964	D52C	INLIN	entrée sans signal.
-10951	D539	GDBUFS	mise à zéro du bit 7 de tous les catères enregistrés par INLIN.
-10925	D553	INCHR	entrée d'un caractère dans l'acc. et mise à zéro du bit 7.

3 - Analyse des données (dans la zone du texte du programme)

177	B1	CHRGET	chargement dans l'accumulateur du caractère pointé par ($B8,$B9) ou TXTPTR (suivant ou actuel) et discrimination du type C = 0 pour un chiffre Z = 1 pour la fin d'une ligne ou d'une instruction.
183	B7	CHRGOT	
-9716	DAØC	LINGET	chargement dans $50,$51 ou LINNUM du numéro de la ligne pointée par $B8, $B9 ou TXTPTR et poursuite de l'analyse.
-6411	E6F5	GTBYTC	saisie d'un caractère par CHRGET et évaluation à partir de TXTPTR pour X.
-6408	E6F8	GETBYT	évaluation de l'expression pointée par TXTPTR et mise du résultat dans FAC.
-6405	E6FB	CONINT	puis FAC→entier ≤255 dans X et FACLO.
-10726	D61A	FNDLIN	recherche dans le programme l'adresse de l'instruction dont le n° est dans LINNUM ($50,$51). Si C = 1, résultat dans LOWTR ($9B,$9C) sinon LOWTR pointe à l'instruction de n° le plus élevé.
-6234	E74C	COMBYTE	vérifie que TXTPTR pointe sur une virgule et continue l'analyse par GETBYT.

(suite)

-6330	E746	GETNUM	saisie d'un nombre pour évaluation et test virgule sur caractère suivant.
-8857	DD67	FRMNUM	évalue une expression pointée par TXPTR et met le résultat en FAC en s'assurant que c'est un nombre.
-6318	E752	GETADR	FAC→entier (2 octets) en $50,51.
-2087	F7D9	GETARYPT	recherche d'une variable par son nom pointé par TXTPTR et
-8221	DFE3	PTRGET	résultat dans $83,$84 ou VARPNT et Y,A - si elle n'existe pas déjà, **création** - la place du nom dans la table des variables est en $9B,9C ou LOWTR.
-8067	EØ7D	ISLETC	l'acc a contient-il un code ASCII d'une lettre, oui C = 1.
-8526	DEB2	PARCHK	vérification des parenthèses
-8520	DEB8	CHKCLS	TXTPTR pointe-t-il sur) ?
-8517	DEBB	CHKOPN	TXTPTR pointe-t-il sur (?
-8514	DEBE	CHKCOM	TXTPTR pointe-t-il sur , ?
-8512	DECØ	SYNCHR	sinon, erreur de syntaxe si oui l'analyse continue.
	EC4A	FIN	enregistre le nombre flottant pointé par TXTPTR dans FAC.

4 - Affichage de données

-4818	ED2E	PRNTFAC	affiche le FAC ($9D-A2) et le détruit.
-9414	DB3A	STROUT	affiche le chaîne pointée par (Y,A).
-9411	DB3D	STRPRT	affiche la chaîne pointée par (FACMO,FACLO).
-9385	DB57	OUTSP	affiche un espace.
-9477	DAFB	CRDO	retour chariot.
-9382	DB5A	OUTQST	?
-9380	DB5C	OUTDO	affiche l'acc A avec les modes I,F,N.
-4839	ED19	INPRT	affiche "IN" n° de ligne courante.
-4828	ED24	LINPRT	affiche un entier dans X,A.
-9515	DAD5	PRINT	instruction d'affichage.

5 - Arithmétique et fonctions algébriques

CONSTANTES NUMERIQUES

EØFE	9Ø	8Ø	ØØ	ØØ	2Ø	-2^{16} = -32767.0005
EDØA	9B	3E	BC	1F	FD	99 999 999,9
EDØF	9E	6E	6B	27	FD	999 999 999
ED14	9E	6E	6B	28	00	1 000 000 000 = 10^9

(suite)

FØ66	81	49	ØF	DA	A2	PI/2 = 1.57079633
FØ6B	83	49	ØF	DA	Ø2	2xPI = 6.28318531
FØ7Ø	7F	00	00	00	00	1/4
EE64	80	00	00	00	00	1/2
E913	81	00	00	00	00	1
E92D	80	35	04	F3	34	SQR(0 5) = 0.707106781
E932	81	35	04	F3	34	SQR(2) = 1.41421356
E937	80	80	00	00	00	-1/2
E93C	80	31	72	17	F8	LOG(2) = .693147181
EA50	84	20	00	00	00	10
EEDB	81	38	AA	3B	2A	LOG(e)/LOG(2) = 1.44269504

FONCTIONS

E7A0	FADDH	(FAC) ← (FAC) + 1/2
E7A7	FSUB	ARG ← (Y,A) et appel FSUBT
E7AA	FSUBT	FAC ← ARG-FAC
E7BE	FADD	ARG ← (Y,A) et appel FADDT
E7C1	FADDT	FAC ← FAC + ARG
E941	LOG	FAC ← LN(FAC)
E97F	FMULT	ARG ← (Y,A) et appel FMULTT
E982	FMULTT	ARG ← FAC * ARG
E9E3	CONUPK	ARG ← (Y,A)
EA39	MUL10	FAC ← FAC * 10
EA55	DIV10	FAC ← FAC/10
EA66	FDIV	ARG ← (Y(A) et appel FDIVT
EA69	FDIVT	FAC ← ARG/FAC
EB80	SGN	FAC ← signe de FAC
EB82	SIGN	A ← signe de FAC (1 si >0, Ø si Ø, FF si <0)
EB93	FLOAT	FAC ← A devient flottant.
EBAF	ABS	valeur absolue FAC ← FAC.
EC23	INT	plus grande valeur entière inférieure FAC ← FAC.
EBF2	QINT	plus grande valeur entière inférieure si FAC < 32767.
E10C	AYINT	plus grande valeur entière inférieure dans mantisse FAC.

6 - Fonctions sur les chaînes de caractère

DEC	*HEX*		
- 8837	DD7B	FRMEVL	évaluation d'une expression à partir de TXTPTR.
- 8575	DE81	STRTXT	TXPTR → Y,X puis appelle STRLIT
- 7193	E3E7	STRLIT	met un caractère de fin de chaîne en ENDCHR.

- 7187 E3ED STRLT2 construit un descripteur de chaîne en DSCTMP, FACMO, LO et conduit à PUTNEW.
- 7126 E42A PUTNEW range DSCTMP dans un descripteur temporaire pointé par FACMO, LO.
- 7203 E3DD STRSPA conduit à GETSPA et range le pointeur et la longueur en DSCTMP.
- 7086 E452 GETSPA libère de l'espace pour une chaîne en déplaçant FRESPC et FRETOP vers le bas
  - peut émettre "OUT OF MEMORY"
  - met à jour DSCTMP.
- 6761 E597 CAT concaténation de la chaîne décrite par (FACMO, LO) et celle pointée par TXTPTR + 1.
- 6686 E5E2 MOVSTR déplace la chaîne pointée par Y,X et A de longueur dans la position pointée par FRESPC (\$71,\$72).
- 6700 E5D4 MOVINS déplace la chaîne dont le descripteur est pointé par STRNF1 vers FRESPC
- 6659 E5FD FRESTR vérifie que FAC adresse une chaîne et conduit à FREFAC.
- 6656 E600 FREFAC libère l'espace occupé par une chaîne temporaire.
- 6603 E635 FRETMS libère le descripteur temporaire sans libérer la chaîne.
- 7036 E484 GARBAG récupération de l'espace occupé par les chaînes abandonnées et déplacement vers le haut des autres.

VARIABLES UTILISEES PAGE ZERO

DEC	*HEX*		
13	D	CHRAC = "	
14	E	ENDCHR = ØØ	
17	11	VALTYP = 1	si chaîne dans FAC.
82	52	TEMPPT	pointeurs temporaires.
83	53	LASTPT	pointeurs temporaires.
	5E, 5F	INDEX	pointeurs temporaires.
111, 112	6F, 70	FRETOP	bas de la zone chaîne.
113, 114	71, 72	FRESPC	fin de la zone libre.
133, 134	85, 86	FORPNT	utilisé par COPY pour libérer de l'espace.

DEC	*HEX*		
148,149	94,95	HIGHDS	utilisés par BLTU
150,151	96,97	HIGHTR	pour l'adresse de destination.
155,156	9B,9C	LOWTR	utilisé par BLTU.
157,158,159	9D,9E,9F	DSCTMP	descripteur de chaîne.
160,161	AØ,A1	FACMO,LO	pointeur de descripteur.
171,172	AB,AC	STRNG1	pointeur utilisé par MOVINS.
173,174	AD,AE	STRNG2	pointeur utilisé par STRLT2.

7 - Fonctions graphiques haute résolution

ADRESSES

page 1 : $2ØØØ - $3FF7
page 2 : $4ØØØ - $5FF7

ligne Ø : $2ØØØ - $2Ø27 (40 octets / ligne)
ligne 64 : $2Ø28 - $2Ø4F
ligne 128 : $2Ø5Ø - $2Ø77
ligne 8 : $2Ø8Ø - $2ØA7
ligne 72 : $2ØA8 - $2ØCF
ligne 136 : $2ØDØ - $2ØF7

lignes 16,80,144,24,88,152 : $21ØØ - $21F7
lignes 32,96,160,40,104,168 : $22ØØ - $22F7
lignes 48,112,176,56,120,184 : $23ØØ - $23F7

Soit n le n° de la ligne et $a son adresse des lignes précédemment listées alors :

la n+1e ligne a pour adresse $a + $4ØØ
la n+2e ligne a pour adresse $a + $8ØØ
la n+3e ligne a pour adresse $a + $CØØ
la n+4e ligne a pour adresse $a + $1ØØØ
la n+5e ligne a pour adresse $a + $14ØØ
la n+6e ligne a pour adresse $a + $18ØØ
la n+7e ligne a pour adresse $a + $1CØØ

VARIABLES DE LA PAGE ZERO

DEC	*HEX*	*Nom*	*Rôle*
26,27	$1A,1B	SHAPE L,H	pointeur dans la table des formes.
28	$1C	HCOLOR1	dépend de la parité de l'abscisse x du HMASK et HCOLORØ.
29	$1D	COUNT H	compteur dans le tracé de ligne.
38,39	$26,$27	HBASL,H	adresse du début d'une ligne y.

DEC	HEX	Nom	Rôle
48	$3Ø	HMASK	$81, $82, $84, $88 b0, b1, b2, b3 $9Ø, $AØ, $CØ b4, b5, b6
82	$52	DY	incrément de y pour HLINE.
83	$53	QDRNT	angle de rotation pour DRAW.

VARIABLES DE LA PAGE ZERO UTILISEES PAR LES FONCTIONS GRAPHIQUES

DEC	HEX	Nom	Rôle
224,225	$EØ,$E1	xH,xL	coordonnées écran du point
226	$E2	y	tracé par HPLOT.
228	$E4	HCOLORØ	ØØ, 2A, 55, 7F noir, vert, violet, blanc 8Ø, AA', D5, FF noir, orange, bleu, blanc
229	$E5	XD7	n° d'octet dans une ligne pour le point d'abscisse x.
230	$E6	HPAG	$2Ø : page 1 $4Ø : page 2
231	$E7	SCALE	facteur d'échelle d'une forme.
232,233	$E8,$E9	SHPTAB	pointeur du début de la table de formes.
234	$EA	CC	indicateur de collision entre deux formes ; est remis à zéro par DRAW.

FONCTIONS GRAPHIQUES

DEC	HEX	Nom	Rôle, Résultat
-3112	F3D8	HGR2	effacement des pages.
-3102	F3E2	HGR	$2Ø→HPAG. $4Ø→HPAG.
-3082	F3F4	BKGND	écran d'une couleur uniforme couleur→acc→HCOLOR1.
	F6FØ	HCOLOR	couleur→X→HCOLORØ.
-3055	F411	HPOSN	x→y,x→xH xL y→acc→y calcul de HBASL,H ; HMASK et XD7 et HCOLOR1.

FONCTIONS GRAPHIQUES (suite)

DEC	*HEX*	*Nom*	*Rôle, Résultat*
-2971	F465	INTX	suivant, incrémentation ou décrémentation de xH, xL et y.
-2861	F4D3	INTY	suivant, incrémentation ou décrémentation de y→HBASL,H.
-2613	F5CB	IPOS	HBASL,H ; XD7→xH, xL et y.
-2985	F457	HPLOT	tracé du point x→y,x et y→A avec le secours de HPOSN et PLOT.
-2982	F45A	PLOT	tracé du point défini par HCOLOR1 ; HMASK ; XD7→Y ; HBASL,H suivant les instructions : LDA HCOLOR1 EOR (HBASL),Y AND HMASK EOR (HBASL),Y STA (HBASL),Y
-2758	F53A	HLINE	tracé d'une ligne quelle que soit sa direction du point actuel au point x→X,A et y→Y.

Dec	*Hex*	*Nom*	*Rôle*
-15360	C400	**DEBUT**	Début du sous-programme de gestion Souris.
-14464	C780	**XMBASIC**	Point d'entrée depuis Basic, reconnaissance de PR£4 suivi de CHR$(1) ou CHR$(Ø) ; appel de XSETMOU et XMHOME.
-14433	C79F	**BASICIN**	Exécute INPUT X,Y,S en appelant XMREAD, HEXTODEC et en analysant MOUSTAT. Rend S négatif si une touche du clavier est enfoncée. Met S à 1 si le bouton est enfoncé. Met S à 2 si le bouton vient d'être enfoncé. Met S à 3 si le bouton vient d'être relâché. Met S à 4 si le bouton est relâché.
-15332	C41C	**INITMOUSE**	Remise à zéro de MOUSTAT, MINXH.L, et MAXXH.L=MAXYH.L=$3FF ; appel de XMHOME et de XSETMOU Acc=Ø.
-15299	C43D	**XSETMOU**	Le mode opératoire est dans A et est contrôlé (C=1 si non valide), puis chargé dans MOUMODE. Le bit I est mis à Ø pour autoriser les interruptions si nécessaire. Le mode d'interruptions pour SETIOU vaut $55.
-15276	C454	**SETIOU**	Le mode d'interruption entré par l'accumulateur est décodé pour modifier les états des commutateurs logiciels.
-15253	C46B	**POSMOUSE**	Met à jour MOUXH.L et MOUH.L par le biais de XMINT et met C à Ø
-15251	C46D	**XMHOME**	MOUSTAT à zéro, MINXH.L dans MOUXH.L et MINYH.L dans MOUYH.L "place" la souris dans le coin supérieur le plus à gauche.
-15228	C484	**XMCLEAR**	Met à zéro MOUXH.L, MOUYH.L et MOUARM et C.
-15211	C495	**XMREAD**	Détecte si la Souris a bougé en testant MOUARM et MOUBUT et MOUSTAT, puis met à jour MOUSTAT. Retourne avec C=Ø.
-15184	C4BØ	**XMCLAMP**	Enregistrement de nouvelles valeurs MAXH.L et MINH.L dans MAXXH.L, MINXH.L si A=Ø et dans MAXYH.L, MINYH.L si A=1.

Dec	*Hex*	*Nom*	*Rôle*
-151Ø8	C4CF	**XMINT**	Prise en charge de la demande d'interruptions. MOUX1 et MOUY1 sont enregistrés dans les registres X et Y puis :
-15147	C4D5	**MOUSEINT**	Les bits 7, 6, 5 de MOUSTAT sont mis à Ø. Teste VBLINT, si oui appel de CHKMOU ; si VBLINT n'est pas l'origine, teste MOUARM, MOUBUT et MOUMODE avant CHKMOU.
-151Ø8	C4FC	**XMTSTINT** ou **SERVEMOUSE**	Indique par C si l'origine de l'interruption n'est pas la Souris. Empile l'accumulateur. Met C à Ø avant de tester MOUSTAT avec MISTAT.
-15119	C4F1	**MISTAT**	Teste les bits 3, 2, 1 de MOUSTAT. Met C à 1 s'ils sont tous nuls.
-15Ø64	C528	**CHKMOU**	Teste MOUXINT et MOUYINT puis met à jour MOUXH.L et MOUYH.L en tenant compte des directions et des limites. Puis stimule MOUCLR et ENVBL si nécessaire et met à jour MOUARM et MOUSTAT.

Codification du mode opératoire pour XSETMOU

L'accumulateur sera chargé dans MOUSMODE.

Contenu	*Mode d'opération*
ØØ	La Souris est hors-ligne.
Ø1	Transparent. Les interruptions ne sont mises à profit que pour mettre à jour les paramètres de la Souris, mais le programme doit lire lui-même ces paramètres. Ce mode est le seul utilisable en Basic.
Ø3	Interruptible par les mouvements.
Ø5	Interruptible par le bouton.
Ø7	Interruptible par les mouvements ou le bouton.
8Ø	La Souris est hors-ligne, VBLINT est armé.
Ø9	Transparent avec VBLINT armé.
ØB	Interruptible par les mouvements et par VBLINT.
ØD	Interruptible par le bouton et par VBLINT.
ØF	Interruptible par les mouvements, le bouton et VBLINT.

Codification des modes d'interruption pour SETIOU

L'accumulateur indique quels sont les bits à changer.

Contenu	*Interruption*
D7	Front descendant de YØ.
D6	Front montant de YØ.
D5	Front montant de XØ.
D4	Front descendant de XØ.
D3	Autorise VBLINT.
D2	Masque VBLINT.
D1	Autorise la Souris à interrompre.
DØ	Masque les interruptions venant de la Souris.

ADRESSES SOURIS

Dec	*Hex*	*Nom*	*Rôle*
1528,1272	5F8,4F8	MAXH,L	Nouveau maximum des coordonnées.
14ØØ,1144	578,478	MINH,L	Nouveau minimum des coordonnées.
14Ø5,1149	57D,47D	MINXH,L	Minimum en X.
1533,1277	5FD,4FD	MINYH,L	Minimum en Y.
1917,1661	77D,67D	MAXXH,L	Maximum en X.
2Ø45,1663	7FD,67F	MAXYH,L	Maximum en Y.
14Ø4,1148	57C,47C	MOUXH,L	Valeur actuelle de X.
1532,1276	5FC,4FC	MOUYH,L	Valeur actuelle de Y.
1916	77C:	MOUSTAT	Etat de la Souris depuis le dernier mouvement, les bits à 1 signifient :
	b7		Bouton enfoncé.
	b6		Bouton toujours enfoncé depuis la dernière saisie.
	b5		La Souris a bougé depuis la dernière saisie.
	b4		Réservé.
	b3		VBLINT est à l'origine de l'interruption.
	b2		Le bouton est à l'origine de l'interruption.
	b1		Les mouvements ont provoqué l'interruption.
	bØ		Réservé.
2044	7FC:	MOUMODE	Mode opératoire de la Souris.
	b7-b4		Réservés.
	b3		VBLINT est armé (non masqué).
	b2		Le bouton va autoriser VBLINT.
	b1		Les mouvements vont autoriser VBLINT.
	bØ		La Souris est en-ligne.
1660	67C	MOUARM	Autorise les interruptions dues aux mouvements ($Ø2) et au bouton ($Ø4).

Indications	*Etat Ø*	*Etat 1*	*Indicateur Nom*	*Adresse*
Interruptions déclenchées				
par le mouvement horizontal				
par XØ sur son front	Montant	Descendant	RDXØEDGE	CØ42
par le mouvement vertical				
par YØ sur son front	Montant	Descendant	RDYØEDGE	CØ43
Masquages des interruptions dues à XØ et YØ	NON	OUI	RDXYMSK	CØ4Ø
Interruption provoquée				
par XØ	NON	OUI	MOUXINT	CØ15
par YØ	NON	OUI	MOUYINT	CØ17
par VBLINT	NON	OUI	VBLINT	CØ19
Bouton	Enfoncé	Relâché	MOUBUT	CØ63
Signal X1	Ø	1	MOUX1	CØ66
Signal Y1	Ø	1	MOUY1	CØ67
Masquage de VBLINT	NON	OUI	RDVBLMSK	CØ41

LES COMMUTATEURS SOURIS

Effets si le commutateur est à 1	*Mis à Ø*		*Mis à 1*	
	Nom	*Adresse*	*Nom*	*Adresse*
Interruptions déclenchées par les fronts du signal XØ par les fronts du signal YØ		 CØ5C CØ5E		 CØ5D CØ5F
Masquage des interruptions dues à XØ et YØ	MOUENBL	CØ50	MOUDSBL	CØ58
Masquage de l'interruption due à VBLINT	ENVBL	CØ5B	DISVBL	CØ5A
Remise à zéro des demandes dues à XØ et YØ à VBLINT	 MOUCLR VBLCLR	 CØ48 CØ7Ø		

1 - Les positions de la Souris et son état seront disponibles dans les octets MOUXH, MOUXL, MOUYH, MOUYL et MOUSTAT.

2 - Appelez INITMOUSE.

3 - Masquez les interruptions (SEI) pendant l'enregistrement des maxima et minima par XMCLAMP.

4 - Initialisez les positions avec XMHOME ou XMCLEAR ou POSMOUSE.

5 - Mettez le mode opératoire dans l'accumulateur et appelez XSETMOU.

6 - Si vous avez choisi un mode interruptible, SERVEMOUSE sera appelé et détectera la source de l'interruption. C vaut 1 si ce n'est pas la Souris.

7 - Masquez les interruptions et appelez XMREAD. Lisez les nouvelles valeurs de coordonnées et d'état et autorisez à nouveau les interruptions.

BOOT : Mise en place du SED (et démarrage à froid)

Programmes	*Localisation*	*Occupation*	*Rôle*
1 - BOOT Ø	PROM carte-contrôleur $C6ØØ	256 octets	charge BOOT 1 en MEV
2 - BOOT 1	DISQUETTE : piste Ø, secteur Ø MEV : $8ØØ-$9ØØ	1 secteur 256 octets	charge BOOT 2 et lui-même
3 - BOOT 2	DISQUETTE : piste Ø, s. 1 à 9 MASTER ↙ ↘ SLAVE (48 K) $37ØØ-$4ØØØ $B7ØØ-$CØØØ	9 secteurs 2304 octets	contient RWTS charge le SED et éventuelle- ment le trans- lateur
BOOT 1	DISQUETTE : piste Ø, secteur Ø MASTER ↙ ↘ SLAVE (48 K) $36ØØ-$36FF $B6ØØ-$B6FF	1 secteur 256 octets	version de BOOT 1 dispo- nible pour ini- tialiser une disquette vierge
4 - SED	DISQUETTE : piste 2, sect. 4 à Ø piste 1, sect. F à Ø piste Ø, sect. F à C MASTER ↙ ↘ SLAVE (48 K) $1DØØ-$36ØØ $9DØØ-$B6ØØ	25 secteurs 6400 octets	système d'ex- ploitation des commandes et de gestion de l'espace sur disquette
Translateur (relocator)	DISQUETTE MASTER piste Ø, sect. A et B MEV : $1BØØ-$1DØØ (n'existe pas dans une disquette) SLAVE	2 secteurs 512 octets	réinstalle le SED à sa place définitive $9DØØ-$CØØØ (48 K)

Organisation d'une disquette

Une disquette est constituée de 35 pistes et les données sont transmises par secteur de 256 octets.

Version 3.3
16 secteurs/piste 560 secteurs/disquette dont 496 utiles
soit 126976 octets utiles

Occupation des pistes et secteurs

Pistes Ø, 1, 2	: SED	(système d'exploitation)
Piste $11, secteur Ø	: VTOC	(occupation)
Piste $11, secteurs $F à $1	: DIRECTORY	(catalcg)
Pistes $12 à $22 et	: programmes et fichier	
$16 à $3	: utilisateurs	

(Le fichier le plus long enregistrable sur une disquette a environ 126000 octets).

Le "directory" peut gérer un maximum de 105 références.

Une référence est un ensemble de 35 octets comprenant :
- l'adresse de la liste des secteurs occupés (n° de piste, n° de secteur) par le fichier référencé,
- le type de fichier A, I, T, B, verrouillé ou non,
- le nom du fichier (30 caractères),
- la longueur en nombre de secteurs occupés (2 octets).

Commandes par ordre d'apparition dans la table des commandes

COMMANDES SED

Index			ADRESSE D'ENTREE HEX	DEC
$00	0	INIT	$A54F	42319
$01	1	LOAD	$A413	42003
$02	2	SAVE	$A397	41879
$03	3	RUN	$A4D1	42193
$04	4	CHAIN	$A4F0	42224
$05	5	DELETE	$A263	41571
$06	6	LOCK	$A271	41585
$07	7	UNLOCK	$A275	41589
$08	8	CLOSE	$A2EA	41706
$09	9	READ	$A51B	42267
$0A	10	EXEC	$A5C6	42438
$0B	11	WRITE	$A510	42256
$0C	12	POSITION	$A5DD	42461
$0D	13	OPEN	$A2A3	41635
$0E	14	APPEND	$A298	41624
$0F	15	RENAME	$A281	41601
$10	16	CATALOG	$A56E	42350
$11	17	MON	$A233	41523
$12	18	NOMON	$A23D	41533
$13	19	PR#	$A229	41513
$14	20	IN#	$A22E	41518
$15	21	MAXFILES	$A251	41553
$16	22	FP	$A57A	42362
$17	23	INT	$A59E	42398
$18	24	BSAVE	$A331	41777
$19	25	BLOAD	$A35D	41821
$1A	26	BRUN	$A38E	41870
$1B	27	VERIFY	$A27D	41597

Localisation	*S.E.D.*	*point d'entrée*
$B600-B6FF	RWTS	$B7B5
$AAC9-B5FF	gestion des commandes	$AAFD
$9D00-AAC8	programme principal	$9D00
$9600-9CFF	3 buffers de 595 octets	

Configuration 48K MEV

RWTS (Read - Write - Track - Sector)
(Lecture - Ecriture - Piste - Secteur)

Sous-programme d'accès à 1 secteur : RWTS

Table des paramètres : IOB

L'adresse de IOB est à charger dans les registres A (poids forts) et Y (poids faibles) avant d'appeler RWTS :

Exemple :

```
LDA  #$10
LDY  #$00
JSR  $3D9
RTS

* 1000 : 01 60 01 00 11 0C 11 10
* 1008 : 00 09 00 00 01
* 1011 : 00 01 EF D8
```

IOB : octet n° 4 : n° de la piste ($11)
octet n° 5 : n° du secteur ($0C)
octets 6, 7 : adresse de la DCT ($1011)
octets 8, 9 : adresse de la zone de transfert en MEV ($900)
octet C : code de la commande
00 positionnement
01 lecture
02 écriture
03 formatage

DCT constantes : 00 01 EF D8 du périphérique

Adresse Hex	*Contenu*	*Rôle*
3DØ	JMP $9DBF	redémarrage à chaud.
3D3	JMP $9D84	démarrage à froid.
3D6	JMP $AAFD	gestion des commandes.
3D9	JMP $B7B5	lecture-écriture d'1 secteur (RWTS).
3DC	LDA $9DØF LDY $9DØE RTS	recherche de l'adresse de la liste des paramètres pour la gestion des commandes.
3E3	LDA $AAC2 LDY $AAC1 RTS	recherche de l'adresse de la table IOB des paramètres de RWTS.
3EA	JMP $A851	pour remplacer les vecteurs d'E/S $38, $39 et $36, $37 avec les pointeurs du SED.
3EF	JMP $FA59	en cas de BRK en MONITOR.
3F3,3F2	$9DBF	adresse de renvoi en cas de 'Reset' (SOFTEV).
3F4	$38	PWRUP = ($3F3) ⊕ $A5.
3F5	JMP $FF58	en cas de &.
3F8	JMP $FF65	en cas de 'CTRL Y'.
3FB	JMP $FF65	si interruption non masquée.
3FE	$FF65	si interruption.

Adresses diverses du SED en MEV

- Fichier binaire chargé en MEV par BLOAD
  adresse en $AA72, AA73
  longueur en $AA6Ø, AA61
- Programme exécuté au démarrage à froid
  nom : piste 1, secteur 9, octet $75...
  type * 9E42 : 34 BRUN
  * 9E42 : 14 EXEC
- Retrait de la pause durant CATALOG
  AE34 : 6Ø

SYSTEM MASTER 3.3

B	FID	- recopie de programmes ou fichiers. - taux d'occupation de la disquette.
A I	COPY	- recopie d'une disquette entière (avec 1 ou 2 lecteurs).
B	MASTER CREATE	- création d'une disquette MASTER à partir d'une disquette déjà initialisée et utilisable mais de type SLAVE. - possibilité de changer le nom du programme qui s'exécute au démarrage.

DAKIN5 programming aids 3.3

the Patcher	visualisation et modification de n'importe quel secteur.
the Peeker	lecture d'un fichier.

Implantation des fichiers et programmes sur une disquette

Exemple :

1 - extrait du CATALOG

```
DISK VOLUME 254
 (T)(n)  (NOM)
*A 006 HELLO
*I 018 ANIMALS
*T 003 APPLE PROMS
*I 006 APPLESOFT
*I 026 APPLEVISION
*I 017 BIORHYTHM
*B 010 BOOT13
```

* verrouillé en écriture ; (T)A,I,T, B type
n : nombre de secteurs occupés
NOM du fichier

2 - extrait du DIRECTORY ou répertoire de la disquette

Piste $11 Secteur $0F

```
00- 00<11 0E>00 00 00 00 00 .........
08- 00 00 00[13 0F](82)/C8 C5 ......HE
10- CC CC CF A0 A0 A0 A0 A0 LLO
18- A0 A0 A0 A0 A0 A0 A0 A0
20- A0 A0 A0 A0 A0 A0 A0 A0
28- A0 A0 A0 A0/[06 00][14 0F]    ....
30-(81)/C1 CE C9 CD C1 CC D3 .ANIMALS
38- A0 A0 A0 A0 A0 A0 A0 A0
40- A0 A0 A0 A0 A0 A0 A0 A0
48- A0 A0 A0 A0 A0 A0 A0/[12       .
50- 00[15 0F](80)/C1 D0 D0 CC ....APPL
58- C5 A0 D0 D2 CF CD D3 A0 E PROMS
60- A0 A0 A0 A0 A0 A0 A0 A0
68- A0 A0 A0 A0 A0 A0 A0 A0
70- A0 A0/[03 00][16 0F](81)/C1   .....A
78- D0 D0 CC C5 D3 CF C6 D4 PPLESOFT
80- A0 A0 A0 A0 A0 A0 A0 A0
88- A0 A0 A0 A0 A0 A0 A0 A0
90- A0 A0 A0 A0 A0/[06 00][17     ...
98- 0F](81)/C1 D0 D0 CC C5 D6 ..APPLEV
A0- C9 D3 C9 CF CE A0 A0 A0 ISION
A8- A0 A0 A0 A0 A0 A0 A0 A0
B0- A0 A0 A0 A0 A0 A0 A0 A0/
B8-[1A 00][18 0F](81)/C2 C9 CF .....BIO
C0- D2 C8 D9 D4 C8 CD A0 A0 RHYTHM
C8- A0 A0 A0 A0 A0 A0 A0 A0
D0- A0 A0 A0 A0 A0 A0 A0 A0
D8- A0 A0 A0/[11 00][19 0F](84)   .....
E0-/C2 CF CF D4 B1 B3 A0 A0 BOOT13
E8- A0 A0 A0 A0 A0 A0 A0 A0
F0- A0 A0 A0 A0 A0 A0 A0 A0
F8- A0 A0 A0 A0 A0 A0[0A 00]       ..
```

< > piste, secteur de la suite du répertoire

[ ] piste, secteur de la liste des secteurs occupés par ce fichier

○ type de fichier
82 Applesoft
81 Integer
80 fichier T
84 binaire

/ / nom du fichier complété à 30 caractères par des espaces codés A0

___ longueur du fichier en nombre de secteurs

3 - extrait du VTOC ou table d'occupation des secteurs

Piste $11   Secteur Ø

```
00- 04 [11 0A] 03 00 00 FE 00
08- 00 00 00 00 00 00 00 00
10- 00 00 00 00 00 00 00 00
18- 00 00 00 00 00 00 00 00
20- 00 00 00 00 00 00 00 7A
28- 00 00 00 00 00 00 00 00
30- 0D FF 00 00 23 10 00 01
38- '00 00 00 00"00 00 00 00'
40- '00 00 00 00"00 00 00 00'
48- '00 00 00 00"00 00 00 00'
50- '00 00 00 00"00 00 00 00'
58- '00 00 00 00"00 0F 00 00'
60- 'FF FF 00 00"00 00 00 00'
68- '00 7F 00 00"01 FF 00 00'
70- '00 00 00 00"00 00 00 00'
78- '00 00 00 00"00 00 00 00'
80- 'FF E0 00 00"00 00 00 00'
88- '00 00 00 00"00 00 00 00'
90- '00 00 00 00"00 00 00 00'
98- '00 00 00 00"00 00 00 00'
A0- '00 00 00 00"00 03 00 00'
A8- '00 00 00 00"00 00 00 00'
B0- '00 00 00 00"00 00 00 00'
B8- '00 00 00 00"00 00 00 00'
C0- '00 00 00 00'00 00 00 00
```

[ ] piste secteur du 1er secteur du répertoire.

version du SED (3.3)

n° de volume (254.

7A = 122 secteurs max. dans une liste des adresses (piste secteur) des secteurs occupés par un fichier.

23 = 35 pistes maximum par disquette.

10 = 16 secteurs par piste.

ØØ Ø1 = 256 octets par secteur.

' ' secteurs occupés dans chaque piste successive.

FEDCBA98	76543210	

si un secteur est libre le bit correspondant est à 1.

FF FF ØØØØ la piste n° 1Ø a tous ses secteurs libres.

ØØ 7F ØØØØ la piste n° 12 a ses secteurs F,E,D,C,A,9,8,7 occu occupés.

4 - extrait d'une liste d'adresses (piste, secteur) des secteurs occupés par un fichier

Exemple : HELLO, piste $13, secteur $F

```
00- 00 00 00 00 00 00 00 00
08- 00 00 00 00 13 0E 13 0D
10- 13 0C 13 0B 13 0A 00 00
18- 00 00 00 00 00 00 00 00
```

adresses des secteurs successivement occupés par HELLO.

5 - *extrait du* code *d'un programme enregistré sur disquette*

Exemple : HELLO, écrit en langage APPLESOFT

piste $13 secteur $E

```
00-[71 04]/19 08 0A 00 B2 20 ......2
08- 20 2D 2D 20 44 4F 53 20  -- DOS
10- 33 2E 33 20 48 45 4C 4C 3.3 HELL
18- 4F 00/20 08 14 00 B2 20 O. ...2
20- 00/28 08 1E 00 89 3A BA .(....::
28- 00/2E 08 28 00 97 00/59 ...(...Y
30- 08 32 00 BA 22 44 4F 53 .2.:"DOS
38- 20 56 45 52 53 49 4F 4E  VERSION
40- 20 33 2E 33 20 20 20 20  3.3
48- 20 20 20 20 20 20 20 20
50- 30 38 2F 32 35 2F 38 30 08/25/80
58- 22 00/8B 08 3C 00 BA 3A "...<.::
60- BA 22 41 50 50 4C 45 20 :"APPLE
68- 49 49 20 50 4C 55 53 20 II PLUS
70- 4F 52 20 52 4F 4D 43 41 OR ROMCA
78- 52 44 20 20 20 53 59 53 RD    SYS
```

[pf pF] longueur du programme en nombre d'octets de MEV occupés.

pf : poids faibles.
pF : poids forts.

/ / instructions du programme.

6 - *extrait du* code *d'un programme en* binaire

Exemple : BOOT13, piste $19, secteur $E

```
00-[00 17][F0 08]/20 E3 03 84
08- 00 85 01 A0 01 B1 00 8D
10- 90 17 C8 B1 00 8D 91 17
18- 20 58 FC A0 FF C8 B9 96
20- 17 08 09 80 20 ED FD 28
28- 10 F3 A9 BF 85 33 20 6A
30- FD AD 00 02 C9 8D F0 0F
38- C9 B1 90 DC C9 B8 B0 D8
40- 0A 0A 0A 0A 8D 82 17 A9
48- 17 A0 81 20 00 1D B0 F7
50- AD FE 16 8D 8A 17 85 13
58- E6 13 AD FF 16 4A 4A 4A
60- 85 10 A9 17 A0 81 20 00
68- 1D B0 F7 EE 8A 17 EE 86
70- 17 AD 86 17 C5 10 F0 EA
78- 90 E8 AD 82 17 AA A9 00
```

[pf pF] longueur du programme en nombre d'octets en MEV.

[pf pF] adresse d'implantation en MEV du début du programme écrit en langage machine.

' instructions du programme en langage machine.

BOOT13 a été sauvegardé sur disquette par la commande
BSAVE BOOT13, A$8FØ, L$17ØØ

Implantation des fichiers sur disquette

7 - extrait du contenu d'un fichier de type T

Exemple : APPLE PROMS, piste $15, secteur $E

```
00- B7 B5 8D C4 C5 CC A0 B1 75.DEL 1
08- B0 B0 B0 AC B1 B2 B5 B0 000,1250
10- 8D D3 C1 D6 C5 A0 D2 C1 .SAVE RA
18- CE C4 CF CD 8D C8 CF CD NDOM.HOM
20- C5 8D D2 D5 CE 8D 00 00 E.RUN...
28- D0 C1 D2 C1 CC CC C5 CC PARALLEL
30- A0 D0 D2 C9 CE D4 AC B2  PRINT,2
38- B5 B6 AC B8 AC B5 B0 B0 56,8,500
40- 8D 00 00 00 00 00 00 00 ........
48- 00 00 00 00 00 00 00 00 ........
50- C3 CF CD CD D5 CE C9 C3 COMMUNIC
58- C1 D4 C9 CF CE D3 AC B2 ATIONS,2
60- B5 B6 AC B8 AC B1 B2 B5 56,8,125
68- B0 8D 00 00 00 00 00 00 0.......
70- 00 00 00 00 00 00 00 00 ........
78- A8 CE CF D4 A0 C1 D6 C1 (NOT AVA
```

(8D séparateur 'Return'

AC) séparateur ","

enregistrement de longueur fixe.

00 aucune donnée écrite (END OF DATA).

Ce fichier a été ouvert par la commande

OPEN APPLE PROMS, L40

définissant la longueur de chaque enregistrement à 40 caractères.

A partir de l'enregistrement n° 1 l'instruction de lecture des champs est

INPUT N$, BL, BW, ST

et d'écriture

PRINT N$;",";BL;",";BW;",";ST

8 - extrait du contenu d'un fichier séquentiel de type T

```
00- B3 8D D0 D2 C5 CD C9 C5 3.PREMIE
08- D2 8D D3 C5 C3 CF CE C4 R.SECOND
10- 8D D4 D2 CF C9 D3 C9 C5 .TROISIE
18- CD C5 8D 00 00 00 00 00 ME.......
20- 00 00 00 00 00 00 00 00 ........
```

(8D le séparateur 'Return' termine chaque enregistrement dont la longueur est libre.

00 aucune donnée inscrite (END OF DATA).

LE CLAVIER DU NOUVEL APPLE //e

La répartition des touches est au standard AZERTY.

L'interrupteur situé sous le clavier sélectionne le jeu des caractères américains qui diffère sur neuf caractères du jeu français (les crochets ouvrant et fermant, le dièze, l'arrobas, les accolades, la barre à gauche, la barre verticale, le tilde). La répartition des touches reste AZERTY dans ce deuxième jeu, ainsi le crochet droit américain s'obtient en tapant sur la touche § qui a le même code que lui (voir bas de la page 57). Les caractères français qui ne figurent pas dans le jeu américain sont représentés sur le clavier par un double marquage très clair.

La touche SHIFT lock permet la sélection des lettres majuscules, des chiffres et des autres caractères gravés sur le haut des touches (?./+%£*_)comme sur l'Apple //c.

LE MICROPROCESSEUR EST UN 65C02 SUR LES DEUX MACHINES

Le 65C02 comprend en plus des instructions du 6502, les 10 instructions suivantes : BRA, DEA, INA, PHX, PHY, PLX, PLY, STZ, TRB, TSB ainsi que de nouveaux modes d'adressage, l'adressage indirect par la page zéro pour ADC, AND, BIT, CMP, EOR, LDA, SBC, STA et l'adressage absolu indirect indéxé pour l'instruction JMP.

La mémoire interne occupe la zone des adresses C100 à FFFF ; la zone F800 à FFFF est celle où se trouvent les points d'entrée des routines quelle que soit la machine, routines dont les instructions s'étendent dans les zones d'adresses C100 à FFFF.

Dans l'Apple //c, les pages C1, C2, C4, C5, C6 et C7 sont utilisées exclusivement par les routines de gestion des circuits d'interface inclus dans la machine (interfaces séries pour imprimante et modem, interface souris, interface lecteur de disquettes).

Dans l'Apple //e, les pages Cx ont deux usages chacune, la gestion des cartes d'interface branchées sur les connecteurs x (1 à 6) et la gestion interne. Il y a donc par rapport à l'Apple //c des routines supplémentaires dans le nouvel Apple//e qui sont implantées sur les pages C1, C2 et C3 pour la gestion de l'affichage en 80 colonnes et des 128 K de MEV, sur la page C4 pour la gestion des interruptions, sur la page C6 pour l'auto-diagnostic.

Les pages C8 à CF ont un usage multiple sur le nouvel Apple //e, puisque chaque interface peut utiliser cet espace (une seule à la fois) ; la MEM interne occupe aussi cet espace d'adresses. L'Apple //c ne partage pas cet espace.

Adresse	*Nom*	*Description*
F8B7	**TSTROM**	Est-ce qu'il y a de la MEM sur la d'interface connectée en 3 ?
FA47	**NEWBREAK**	En cas de BRK, les MEM d'interfaces seront connectées et l'accumulateur sera sauvé en A5.
FC74	**IRQUSER**	Permet la gestion d'interruption par les MEM d'interfaces.
FC7A	**IRQDONE2**	Restaure l'usage de l'espace C8 à CF et l'état.
FCF0	**GETINST1**	Mini-assembleur, affiche ! avant de retourner en CF9C.
FEC5	**RETCX1**	Restaure l'usage des MEM d'interfaces.
FED7	**SEARCH**	Fonction de recherche d'une chaîne de caractères.
FEF1	**LOOKASC**	Entrée de caractères précédés de '.
C3F4	**IRQDONE**	Point de retour une fois l'interruption traitée.
C3FA	**IRQ**	Point d'entrée principale si interruptions.
C400	**NEWIRQ**	Traitement principal des sources d'interruption.
C4C8	**AMOD1**	Entrée d'un mnémonique et analyse.
C600	**DIAGS**	Test des 128 K de mémoire vive (principale et auxiliaire) et des commutateurs logiciels (sauf INTCXROM).
C803	**BASICINIT**	Routine déclenchée par PR£3.
CF3A	**REL**	Calcul de l'opérande d'un branchement relatif.
CF5F	**DISPLAY**	Affiche l'instruction assemblée.

COMMUTATEURS LOGICIELS SUPPLEMENTAIRES SUR LE NOUVEL APPLE //e

Adresse	*Nom*	*Description*
C006	**SETSLOTCXROM**	Met en ligne les mémoires mortes des interfaces installées dans les connecteurs 1 à 7.
C007	**SETINTCXROM**	Met en ligne les mémoires mortes internes implantées dans les pages C1 à C7.
C00A	**SETINTC3ROM**	Met en ligne les mémoires mortes internes implantées sur la page C3 (affichage 80 colonnes).
C00B	**SETSLOTC3ROM**	Met en ligne la mémoire morte de la carte d'interface installée sur le connecteur 3.
C015	**RDCXROM**	Etat du commutateur de sélection des mémoires mortes d'adresses Cx.
C017	**RDC3ROM**	Etat du commutateur de sélection des mémoires mortes de la page C3.

Adresse	*Nom*	*Description*
C020	**TAPEOUT**	Signal envoyé au lecteur de cassette.
C040	**STROBE**	Signal d'échantillonnage envoyé en sortie.
C058	**SETAN0**	Mise à un de la sortie tout-ou-rien AN0.
C059	**CLRAN0**	Mise à zéro de la sortie tout-ou-rien AN0.
C05A	**SETAN1**	Mise à un de la sortie tout-ou-rien AN1.
C05B	**CLRAN1**	Mise à zéro de la sortie tout-ou-rien AN1.
C05C	**SETAN2**	Mise à un de la sortie tout-ou-rien AN2.
C05D	**CLRAN2**	Mise à zéro de la sortie tout-ou-rien AN2.
C05E	**SETAN3**	Mise à un de la sortie tout-ou-rien AN3.
C05F	**CLRAN3**	Mise à zéro de la sortie tout-ou-rien AN3.
C063	**PB2**	Etat du bouton poussoir n°2.
C067	**PADDL3**	Entrée analogique n°3.
C068 à C06F		Mêmes effets que de C060 à C067.

INTERRUPTIONS

Le vecteur d'interruption en FFFE, FFFF contient l'adresse C3FA dans le nouvel Apple //e. En phase de gestion des interruptions, le //e utilise les pages C8 à CF contenant les routines internes de MEM. Les routines implantées à ces adresses dans les interfaces sont donc déconnectées jusqu'à la fin du traitement principal d'interruptions, mais reconnectées grâce à la variable MSLOT en 7F8 qui contient la valeur Cs, s indiquant le numéro du connecteur en ligne au moment de l'interruption. Le vecteur d'interruption dans l'Apple //c contient C803, et puisque l'espace C8 à CF n'est pas dédoublé dans l'Apple //c, la variable MSLOT n'a plus la même importance qu'en Apple //e.

LE MINI-ASSEMBLEUR

Sur le nouvel Apple //e, la commande ! donnée à partir du Monitor conduit au mini-assembleur qui accepte les instructions en mnémoniques avec la syntaxe suivante :

- une adresse en 4 chiffres hexadécimaux au maximum,
- le caractère :
- un code opération sur 3 lettres appartenant à l'ensemble des 56 mnémoniques du 6502 (attention, ce mini-assembleur ne reconnaît pas les nouveaux mnémoniques du 65C02),
- un opérande pouvant prendre les différentes formes associées à chaque mode d'adressage, mais ne contenant pas de symboles ou d'étiquettes,
- dans le cas d'un branchement relatif, l'opérande est l'adresse de destination.

Après la première instruction, les suivantes sont entrées en remplaçant l'adresse par le caractère espace.

Les erreurs de syntaxe sont signalées par un beep sonore et un caractère ^pour indiquer l'erreur.

Pour quitter le mini-assembleur, il suffit de répondre 'Return' à la sollicitation !

RECHERCHE DE CHAINES DE CARACTERES ASCII

A partir de la sollicitation * du programme Monitor, la commande :

'chaîne de caractères<adr1.adr2 S

recherche la chaîne donnée dans la zone indiquée.

ENTREE DE CARACTERES EN CLAIR

A partir de la sollicitation * du programme Monitor, la commande :

adresse:'x 'y 'z

permet de mémoriser à partir de l'adresse donnée, les codes ASCII (bits 7 à 1) des caractères précédés du caractère apostrophe et suivis du caractère espace.

INDEX

CONSEILS DE LECTURE

Pour faire de la programmation de haut niveau en Basic, et maîtriser le système de l'Apple IIe 6502 ou 65C02, et de l'Apple IIc, P.S.I. vous propose une palette d'ouvrages utiles.

POUR MIEUX PROGRAMMER EN BASIC APPLESOFT

- **Techniques de programmation sur Apple II** - René Belle (Editions du P.S.I.) :
  Pour programmer avec efficacité, voici des routines Basic d'édition, de tri, de recherche ..., réutilisables et adaptables au contexte personnel de l'utilisateur.
- **La pratique de l'Apple II, tome 2** - Nicole Bréaud-Pouliquen (Editions du P.S.I.) :
  Ouvrage consacré au système d'exploitation disque, à la gestion des fichiers, à la carte Apple-clock, etc.
- **Basic plus, 80 routines sur Apple II** - Michel Martin (Editions du P.S.I.) :
  Pour pousser votre Apple au maximum de ses capacités : 80 routines de simulation d'instructions qui n'existent pas en Basic Applesoft.
- **Programmation système de l'Apple II** - Marcel Cottini (Editions du P.S.I.) :
  Pour programmeurs chevronnés sur Apple IIe ou IIc, cet ouvrage donne une bonne connaissance des microprocesseurs 6502 et 65C02 et des centaines d'astuces inédites de programmation.

POUR MAITRISER LE SYSTEME DE L'APPLE II

- **Introduction à ProDOS sur Apple** - Francis Versheure (Editions du P.S.I.) :
  Initiation au système d'exploitation des IIe et IIc, comprenant notamment, la gestion des supports, des catalogues et des fichiers, l'étude d'un système de conversion de DOS à ProDOS, et l'utilisation de l'Applesoft sous ProDOS.

- **Les ressources de l'Apple IIc** - Nicole Bréaud-Pouliquen (Editions du P.S.I.) :
  Une étude poussée du 65C02, des ressources de la mémoire vive, du système d'exploitation ProDOS et de la programmation de la souris.

- **Clefs pour Apple IIe 65C02 et Apple IIc** - Nicole Bréaud Pouliquen (Editions du P.S.I.) :
  Mémento présentant synthétiquement les commandes Basic et système, le jeu d'instructions du 65C02, les messages d'erreur et des routines astucieuses permettant d'utiliser à fond l'ordinateur. Le livre de chevet du programmeur sur AppleII.

A PARAITRE :

- **Système ProDOS sur Apple II** - Marcel Cottini (Editions du P.S.I.) :
  Ouvrage de très haut niveau présentant l'organisation complète de ce système d'exploitation avec de nombreux exemples d'application.

Achevé d'imprimer en février 1986
sur les presses de l'imprimerie Laballery et Cie
58500 Clamecy
Dépôt légal : février 1986
N° d'impression : 601125
N° d'édition : 86595-329-1
ISBN : 2-86595-329-7

Votre avis nous intéresse

Pour nous permettre de faire de meilleurs livres, adressez-nous vos critiques sur le présent ouvrage.

— *Ce livre vous donne-t-il toute satisfaction ?*

....
....
....
....
....

— *Y a-t-il un aspect du problème que vous auriez aimé voir abordé ?*

....
....
....
....
....

Si vous souhaitez des éclaircissements techniques, écrivez-nous, nous ne manquerons pas de vous répondre directement.

Où avez-vous acheté ce livre ?

☐ cadeau ☐ librairie ☐ autres
☐ exposition ☐ boutique micro

Comment en avez-vous eu connaissance ?

☐ publicité ☐ catalogue ☐ autres
☐ exposition ☐ conseils d'un ami

Avez-vous déjà acquis des livres P.S.I. ?

Lesquels ? ....
....
qu'en pensez-vous ? ....
....
....

Nom .... Prénom .... Age..........
Adresse ....
Profession ....
Centre d'intérêt ....

CATALOGUE GRATUIT

Vous pouvez obtenir un catalogue complet des ouvrages PSI, sur simple demande, ou en retournant cette page remplie à votre libraire, à votre boutique micro ou aux

Editions du PSI
BP 86
77402 Lagny-sur-Marne Cedex



Un mémento qui s'ouvre à la bonne page et vous permet d'accéder efficacement à toutes les informations dont vous avez besoin : commandes Basic ou système, messages d'erreur, jeu d'instructions du 65C02, etc.
"Clefs pour Apple //c et //e 65C02" est aussi un recueil d'astuces qui vous apprendront comment justifier un texte à droite, comment modifier la fenêtre d'écran-texte, comment programmer la souris, comment transférer une image en mémoire auxiliaire...
Ces routines vous feront découvrir toute l'originalité de votre Apple //e 65C02 ou de votre Apple //c.

CLEFS POUR APPLE //c et //e 65C02



Éditions du PSI
BP 86
77402 Lagny/Marne
France

ISBN : 2.86595.329.7

P.S.I.

Couverture Mariane de Nayer

Imprimé en France

145 FF